SeDyA: Secure Dynamic Aggregation in VANETs

Rens W. van der Heijden
University of Twente
r.w.vanderheijden@alumnus.utwente.nl

August 10, 2012

Abstract

In Vehicular Ad-hoc Networks (VANETs), the ultimate goal is to let vehicles communicate by exchanging messages through wireless networks to provide safety, traffic efficiency and entertainment applications. Aggregation of information in these messages contributes to this goal by reducing the bandwidth requirements that prevent applications from disseminating messages over a large area. Aggregation will allow applications to exchange high quality summaries of the current status in a specific region, rather than forwarding all individual status messages from this region, increasing the available information for all vehicles.

Most existing work on aggregation in VANETs has neglected to consider security, not providing any guarantees on the data that is collected. Security for aggregates is important because they may be used by other cars for decisions about routing, as well as traffic statistics that may be used in political decisions concerning road safety and availability. The most important challenge for security is that aggregation removes redundancy and the option to directly verify signatures on messages, because multiple messages are merged into one. The few works that discuss secure aggregation are limited because they require roads to be segmented into small regions, beyond which aggregation cannot be performed. The main contribution of this thesis is the introduction of SeDyA, a scheme that allows more dynamic aggregation compared to existing work, while also providing stronger security guarantees for the receiving vehicles.

Acknowledgements

The author thanks Stefan Dietzel and Prof. Frank Kargl for their invaluable guidance and discussions over the course of the thesis. In addition, the author wishes to thank Michael Feiri for his assistance with code development and interesting discussions. The committee members, Prof. Frank Kargl, Prof. Geert Heijink, Dr. Jonathan Petit, and Stefan Dietzel, are thanked for making time in their busy schedules to review this thesis. Finally, the author would like to thank the members of the Distributed and Embedded Security group in the University of Twente and of the Institut für Verteilte Systeme in Ulm University for their hospitality and the office space where most of this thesis was written.

Contents

Chapter 1

Introduction

In recent years, much work has been performed by industry and academia alike to develop Vehicular Ad-hoc Networks (VANETs) to improve safety on the road, as well as introducing information and entertainment applications. A VANET is created by equipping vehicles with an on-board unit (OBU) that is capable of wireless communication, typically using IEEE 802.11p, developed specifically for VANETs. In addition to communication between OBUs, some VANET research envisions the availability of road-side units (RSUs), although in the early phases of VANETs these are expected to be sporadic. RSUs will also use IEEE 802.11p, but will typically also be connected to a back end, and will provide access to the Internet, contact with certificate authorities and the possibility to distribute updates for applications. The greatest challenges of VANETs compared to other types of Ad-hoc Networks include the highly dynamic network conditions, bandwidth constraints and the large amount of vehicles.

For VANETs, security is of essential importance, as attacks on these networks can easily lead to safety risks, in addition to typical security concerns. This gives rise to some types of VANET-specific attacks, including dissemination of false information, large scale privacy violation through tracking and position spoofing. Aside from these attacks, VANETs also pose an additional challenge to security research through its bandwidth and time constraints, requiring novel ideas to provide small signatures, small certificates and efficient signature verification.

Currently, the first VANET protocols are undergoing standardization, including security mechanisms to protect them. The first deployments in real world scenarios are forseen in 5 to 8 years. One of the first of these protocols, for which standardization is nearing completion, is the periodic beaconing service. This protocol defines the basis for many other protocols, as it requires every vehicle to periodically transmit a beacon message to announce its presence, typically with a frequency of 10 Hz. Beacons are strictly single-hop, but will enable many essential safety applications, including brake warnings, collision avoidance and cooperative adaptive cruise control [9,16]. The integrity of the beacon messages, as well as other messages transmitted by the network, will be protected by cryptographic signatures. For these signatures, it is commonly assumed that a public key infrastructure (PKI) specifically designed for VANETs will exist. Since these beacon messages typically include location and speed information, VANETs also raise privacy concerns, especially when one considers a typical X.509-style PKI, where a public key is linked to a person or device. To address this issue, but still satisfy the integrity requirements, pseudonym schemes have been proposed, which in essence provide a vehicle with multiple identities (and thus multiple keys) to use. However, for some applications, such as cooperative adaptive cruise control, it is necessary to link sequences of messages from the same vehicle, to estimate its trajectory. The exchange between different identities and the trade-off between the provided privacy, security and functionality is an active area of research [10, 22].

Some applications, like Internet access and traffic management, require communication beyond the single-hop range that beaconing messages provide. For this purpose, multi-hop communication may be introduced, although it is subject to considerable drawbacks in terms of overhead and message loss due to the dynamic environment. Multi-hop communication may be used to communicate with RSUs, between vehicles, but also to disseminate important information to a specified region. The latter is referred to as geocast, which

allows a single vehicle (or RSU) to transmit messages to a particular region. However, note that in many applications, it is not the information about individual vehicles that is interesting, but instead the situation aggregated over a particular area of the road. Considering the strict bandwidth constraints of VANETs, it is very ineffective to define a scheme where many vehicles disseminate their information to a larger area using geocast techniques. For this reason, recent work has introduced many different applications that apply aggregation in some form. Aggregation allows to compress relevant information about many vehicles in a single message, rather than disseminating all these messages. Due to the ad-hoc nature of VANETs, it is an ideal setting to apply in-network aggregation to perform aggregation in a distributed fashion. Contrary to more traditional approaches, as those employed in sensor networks, the relevant aggregation schemes cannot rely on network structure or a central receiving node. An additional advantage is that not forwarding the individual messages provides slightly more privacy, since messages containing the vehicle's pseudonym are no longer disseminated over a long range, and the information is more restricted.

However, it has been shown [13] that security mechanisms used for normal messages are insufficient for aggregation mechanisms, and that the impact of a successful attack on an aggregate message may have much more impact due to the fact that aggregation tries to compress data into one message as much as possible. Thus, while conserving bandwidth, aggregation without security provides an excellent surface for attacks: the impact of an attack on a single message increases. In addition, aggregation reduces the amount of redundant information in the network, thereby reducing the amount of messages that have to be forged or modified by the attacker. In essence, for aggregation to be secure means that every step in the aggregation process must be protected against tampering by attackers. Compared to normal message integrity, aggregation adds an extra challenge here: after several messages have been aggregated, it should be possible to verify that the aggregation process has been performed correctly. Intuitively, proving that a number of messages from distinct vehicles were used in the computation of an aggregate is difficult. Simple solutions like including all signatures of each vehicle are prohibitive in terms of overhead and do not allow for flexibility during aggregation.

In previous work, several solutions have been proposed to provide secure aggregation in VANETs [20, 24, 25]. These solutions each consider the aspects of aggregation and security at the same time; by intertwining both aspects, it is possible to conserve resources at the cost of a less general solution. Compared to previous work on VANET aggregation that does not consider security [11, 29, 33, 49, 50], these schemes are limited in the quality of the information they provide. This is in part because the schemes that do not consider security are not bound to fixed information that is known before the aggregation procedure begins, unlike the secure aggregation schemes. On the other hand, the secure schemes are bound to fixed information, due to the fact that signatures are still used to guarantee that aggregates are merged correctly, and signatures require fixed information to remain verifiable. In addition to signatures, each of the discussed schemes employs a probabilistic counting mechanism for the aggregation process; FM sketches in [20, 24] and z-smallest in [25]. These probabilistic counting algorithms simplify the counting of distinct elements in a distributed fashion through the use of hash functions, thereby providing duplicate insensitivity, a property due to which they are also used in some VANET aggregation mechanisms that do not consider security. However, due to their probabilistic nature, probabilistic counting algorithms like FM sketches also have a negative effect on data quality.

Based on this work, the following research questions have been put forward, which guided the development of a new scheme:

1. How can we provide data integrity and consistency for in-network aggregation in VANETs?
2. Which guarantees and how much confidence can we gain by using cryptography for data integrity and consistency in VANET aggregation?
3. How can we compare different secure aggregation schemes in terms of accuracy, privacy and security guarantees?
4. How can we ensure that aggregation schemes can scale to a sufficiently large area without loosing too much accuracy?
5. Can we obtain a better trade-off between security and privacy than the current state of the art in VANET aggregation?

This thesis will introduce a new secure VANET aggregation scheme called secure dynamic aggregation (SeDyA), which will build on elements from the current state of the art to provide stronger security guarantees. The contribution of this new scheme is twofold. First, it provides a dynamic definition of the aggregation area by employing probabilistic counting techniques, allowing aggregates to be defined over a large area, reducing the amount of messages required to disseminate the relevant information troughout the network. Second, the scheme aims to provide stronger security guarantees in an efficient manner, by employing novel cryptographic primitives. Compared to related work, it will allow the weakening of several unrealistic[1] assumptions while still providing better security. In addition, SeDyA can provide additional input for trust and consistency mechanisms that verify messages based on content rather than cryptographic validity.

To validate the scheme, this thesis will present extensive simulation results to show that the scheme improves on the state of the art, as well as experiments to motivate several design choices in the scheme. Simulations will be performed using an implementation of SeDyA in the Java-based simulator JiST/SWANS, a state of the art discrete event simulator [3]. The main criteria for analysis are accuracy, feasability, and security.

The master thesis is structured as follows. Chapter 2 will discuss the various issues involved in VANETs, VANET aggregation and their security in detail, followed by a discussion of important related work in Chapter 3. Then, SeDyA will be introduced in two chapters, discussing first the high-level goals and solutions in Chapter 4 and then important details and cryptographic background in Chapter 5. Finally, SeDyA will be evaluated with simulations in Chapter 6 and the master thesis will be concluded in Chapter 7.

[1] It should be noted that not all related work was originally developed for VANETs. The assumptions hold in their respective domains, but not in VANETs.

Chapter 2

Problem Statement

This chapter provides an overview of the issues that the scheme from Chapter 4 aims to solve, as well as a discussion of the assumptions on network conditions and the attacker model. Section 2.1 discusses the latter, while Section 2.2 provides a high level overview of the challenges involved. The assumptions in this chapter are similar to those of related work, which is discussed in Chapter 3; the differences are discussed. Finally, Section 2.3 will conclude with some requirements for VANET aggregation.

2.1 Models

This thesis will use assumptions similar to those in the state of the art; these assumptions are made explicit in this section. The differences with a assumptions in related work are discussed and motivated. Specifically, some of the assumptions can be considered unrealistic and are thus adapted to a more general setting for this thesis. Finally, note that different requirements and use cases for aggregation exist; the aggregation model will make explicit what is assumed for this thesis. However, first the network and attacker models will be discussed, first in a general VANET setting and then indicating the specific challenging for aggregation.

2.1.1 Network model

Current literature assumes a VANET will have a typical communication range of about 300 meters [42], with up to 1000 meters under optimal conditions [1]. It should be possible for the VANET to improve safety and provide services even when relatively few vehicles are equipped with wireless technology, as this will facilitate introduction of VANETs into the real world [23]. The network is very dynamic; most communication is expected to occur over single-hop broadcast, as there is no guarantee that sequences of more than one message can be exchanged between two vehicles. For this reason, clustering and other schemes that require knowledge of the network topology, as is common in sensor networks, are typically avoided, although sensor networks are an important source of inspiration for many VANET protocols.

There are several existing ways to dissiminate information in VANETs, each more appropriate for certain types of applications [40]; the most important three are beaconing, geobroadcast and (in-network) aggregation.

Beaconing uses a periodic single-hop link-layer broadcast message to inform other vehicles of the status of the transmitting vehicle and typically contains at least location, heading, speed and time. Beaconing is used for applications like cooperative awareness, but also for packet routing and many other applications that require knowledge of the vehicles' immediate surroundings.

Geobroadcast (also known as geocast) is used to forward a particular packet over multiple hops to all vehicles in a specific destination area. Note that unlike regular unicast packets, the destination is an area instead of a vehicle. Applications that use geobroadcast typically include those that notify vehicles of an event, like a post-collision notification or a road condition notification.

Aggregation provides information about larger areas of interest or over larger periods of time, such as a stretch of road between two exits on a highway in the past ten minutes. This traffic pattern is a series of messages which have information over a certain area with increasing accuracy. By aggregating the information into very few messages, the state of an entire stretch of road can be described in one packet, rather than by the beacons of all vehicles on this stretch of road. Aggregation is used to collect traffic statistics, detect traffic jams and count free parking spaces. For aggregation to work, it is required that duplicates can be avoided and that the order in which the data is processed is irrelevant to the result. Here, duplicates are defined strictly as processing exactly the same message twice.

The network typically contains on-board units (OBUs), integrated in vehicles and connected to its sensors, and some areas there will be road-side units (RSUs), which allow connectivity to a back end. There is a PKI that provides the OBU with key material, or information to compute valid key material, including a number of pseudonyms and system parameters. Key material is typically loaded onto the OBU through some off-line channel, or possibly through a user's home network. Future deployment of VANETs will most likely involve the deployment of RSUs, to provide additional services such as Internet connectivity, tolling applications or as a simple means of collecting network and traffic statistics. However, the cost to deploy RSUs is prohibitive, especially in the initial stages of VANET deployment, when the penentration rate is low and there is thus little benefit. Larger numbers of RSUs are expected to be deployed only after this initial stage, and only in a limited part of the road network, so a general application should be able to operate without access to infrastructure. However, it is reasonable to assume low-bandwidth, low frequency and high latency contact with some back end (or the Internet), even when RSUs are never deployed, since this can also be achieved through an existing mobile phone network. Finally, high-bandwidth low frequency contact may be possible when the car is at home or under maintainance. A more detailed discussion of deployment possibilities and the PKI can be found in [1, 9, 42].

To preserve the privacy of drivers against corporate tracking, as well as large scale tracking by governments, VANETs will employ the use of pseudonyms, typically generated by a certificate authority. Pseudonyms are alternative identities that are used as the identity in a certificate, of which a vehicle recieves multiple, along with the associated private key. The mechanism is similar to a fairly recent RFC on Traceable Anonymous Certificates[1], although the issuance process for pseudonyms is usually different. Pseudonyms protect the privacy by making the different certificates unlinkable, and allowing a vehicle to change the certificate it uses. In most proposals, it is possible for certificate authorities to resolve conflicts[2] by revoking pseudonymity of a user, when ordered to do so by a court. The goal of pseudonyms is not to provide perfect anonymity, but to provide the same level of protection as when VANETs are not used. This means that typical tracking of individual vehicles, for example by driving behind them, is not something that VANETs need to protect against. To achieve the required level of privacy, pseudonyms should switch in a controlled fashion, such that the vehicles cannot be linked after the switch. This may seem counter-intuitive, but without additional protection it is easy to link two pseudonyms after a switch, if there is no transition, by simply matching the locations contained in different beacons [22, 23].

2.1.2 Attacker model

Most of the attacker model for VANETs is the same as for regular Internet services and wireless networks (ie. Dolev-Yao attacker model [14]), where the attacker carries the message. This type of atttacker model includes passive attacks like wiretapping and active attacks like replay, modification, injection or dropping of packets. For some situations, the Dolev-Yao model is too strong and is weakened by adding an honest majority assumption. On the other hand, the attacker may obtain arbitrary certified keys (up to half the total nodes if an honest majority is assumed), because such keys may be obtained from a vehicle directly, or from after-market devices. However, even when an honest majority exists, note that in the interest of privacy, it may be possible to obtain multiple identities (pseudonyms) for a single device [37], enabling the attacker to break certain majority-based schemes that do not protect against this type of attack. Most

[1] RFC 5636, Traceable Anonymous Certificate (2009), which has experimental status; see `http://tools.ietf.org/html/rfc5636`.

[2] The question of which conflicts is an interesting legal question, but is considered out of scope for this thesis.

papers therefore consider a simple constraint to exclude this type of attack, which may be either a short limit on certificate lifetime or another mechanism.

Because VANETs focus mainly on integrity and availability, passive attackers will be composed mostly of academic researchers and governments or corporations looking to obtain personal data by attacking pseudonym schemes. Confidentiality is not as strong a requirement as in normal networks, because as noted in the network model, most traffic patterns will concern public data. Providing confidentiality for such data does not make sense, especially because the attacker could simply purchase a vehicle and listen to the network, trivially bypassing any attempt at providing confidentiality against outsiders. Potential application-specific confidential data, such as fresh pseudonyms for a vehicle, can always be transmitted using a more general higher-layer security protocol like (D)TLS[3].

Active attackers can inject, replay, modify or drop packets and have the option to jam their transmission range for denial of service (DoS) attacks or to prevent messages from arriving. In addition, it is easy for the attacker to obtain legitimate access to the network, as the only necessary resources are valid key material and proper communication equipment, both of which will be available to anyone. It is expected that most attackers will be active, although they will vary greatly in strength. Their goals will vary from common attacks, like users making some extra space for themselves on the road, to extremely rare attacks, like activists that want to block a road network[4] or terrorists that attempt to cause chain-accidents.

The main challenge of active attackers in this model is that they may be legitimate participants for a long time before they attempt an attack. In aggregation, this challenge is even greater, because the location to which the information applies may not be directly verifiable by the receiver. Thus, active attackers may be able to manipulate the receivers' view of the world by injecting false messages that are indistinguishable from legitmate messages. In addition, there is always a risk of a software bug or damaged sensor that may cause incorrect data to be sent. This data is called faulty data, distinguishing it from attacks, whose injections are refered to as malicious data [38]. Thus, even in a best case scenario, a purely cryptographic solution would only be able to identify incorrectly signed or modified messages; a signature provides authenticity, but not necessarily validity. However, it is possible to compare a message with other messages, such as those with similar time and location, exploiting redundancy to check the consistency of messages from different senders. Since aggregation involves the compression of data in a lossy manner, as well as reducing redundancy, there are additional risks involved, leading to several new types of attacks even when the messages are protected against attacks in the preceding model. These new types of attacks are sybil attacks, inflation and deflation attacks and, specifically for VANETs, the remote impersonation attack.

Sybil attacks are an inherent challenge for a system that relies on majority decisions as well as providing pseudonymity; if an attacker can obtain several identities to protect her privacy, she can also use them to artificially represent multiple nodes. Sybil attacks are a difficult problem in general [5, 10, 15], but for VANETs the additional challenge is that no single party must be able to revoke the pseudonymity of any legitimate user. Recently, schemes have been developed to detect sybil attacks in VANETs through trajectory verification [10] and plausibility checking [5]. However, these solutions rely on RSUs and proximity to the attacker, respectively. For Footprint [10], the focus lies on urban scenarios, where RSUs are likely to be deployed in sufficient quantity. The VANET aggregation scenario adds the challenge of doing this detection remotely and with minimal interaction with RSUs. Unlike assumed by Footprint, VANET aggregation will also occur on highway scenarios. Detecting sybil attacks will play a role in any efficient and secure aggregation mechanism. One way to do this is to get rid of pseudonyms; however, this would mean that identities are bound to a node, removing the desired privacy. As noted previously, an alternative is to use a maximum lifetime, enforced by either an issuance mechanism or simply a tight bound on the certificate lifetime.

In- and deflation attacks specifically aim at influencing the value of an aggregate. This can also be achieved by employing sybil attacks; however, the term in- and deflation attacks is used for attacks that attack the aggregation mechanism, either by modifying their own observation or by generating false aggregates [20]. Some countermeasures exist against this type of attack; schemes that include cryptographic countermeasures,

[3]DTLS stands for Datagram TLS; it is basically TLS-like security for UDP connections, which may be more prevalent in VANETs, although this is speculation.

[4]Here, road network refers to a large number of roads that could not be blocked using simple objects.

such as [20,24], will be reviewed in Section 3.4. Additional countermeasures include other techniques such as plausibility checks [12]: these are considered out of scope as they can be used as complementary mechanisms.
Remote impersonation attack is the attack type that is used to influence the knowledge of a target node or group of nodes about a particular area of interest. The attack can be performed from any location, typically *outside* the aggregation area, as opposed to sybil and in- and deflation attacks. Countrary to these other attacks, remote impersionation attacks do not have the attacker as a legitimate participant. Two elements are essential to a remote impersonation attack; first, the attacker must be able to inject an aggregate, and second, the attacker must artificially specify the aggegation area. This is different from modifying a geocast message transmitted from the aggregation area to the target; the attacker may be at a completely different location sending a similar message.

2.2 VANET Aggregation Overview

Aggregation in VANETs can be seen as an instance of distributed aggregation, with a number of phases that illustrate different steps in the aggregation process. Typically distributed aggregation is considered to have one or very few sink nodes. Sink nodes are interested in certain information and generate queries to which the network responds by aggregating in a specific way, as declared in the query (using a language like SQL) [20, 32, 39]. Each node aggregates the data it receives and forwards it to the sink node, until all the data is aggregated and at the sink node.

However, in the VANET model this is somewhat different: first, most nodes (vehicles) are interested in the result of the aggregation process, instead of just the sink node(s). Second, there is no sink node or set of sink nodes that generate the queries that are to be answered by the network. The first problem could be partially solved by employing a dissemination scheme after first performing aggregation to some sink nodes. However, this does not address the problem of generating queries, nor does it address the problem for a node to determine whether they have a complete aggregate that should be disseminated. Another approach is employing in-network aggregation, a process where the network nodes themselves perform aggregation on the messages they receive as they forward them. In-network aggregation is more *fuzzy*, meaning here that it is harder to ensure that every node participates correctly, but it is more efficient than applying dissemination back nto the network after aggregation has been performed. While in-network aggregation does not solve the lack of sink nodes by definition, it allows for a much simpler solution than ad-hoc selection of sink nodes; the queries may simply be embedded in the machine code of the application. If this solution is used with the other approach, then nodes must still somehow detect that they are a sink node and initiate dissemination.

The issue of generating queries poses a risk of denial of service attacks; allowing arbitrary vehicles to specify arbitrary queries to which other vehicles respond is a recipe for disaster. When these queries are defined within the application, it may be challenging to update them, thus bounding the amount of possible queries. Finally, it is also important to be able to separate the same query for different sections of road or time span. One common solution to this is to use a fixed piece of information related to the aggregate, such as a fixed aggregation area, as the query identifier, allowing to distinguish similar queries. However, this is undesirable from the perspective of application users and developers, as the solution limits the flexibility of the application because it can not dynamically expand the aggregation area [12, 13]. In this work, a scheme to define an aggregation area in a more dynamic fashion, while still retaining the useful property of a unique query identifier, will be introduced.

The theoretical background behind aggregation and its application to sensor networks will be discussed in Section 2.2.2, while secure aggregation in VANETs will be the topic of Section 3.4. The remainder of this section will first motivate the use of aggregation in VANETs, then discuss some aggregation requirements, propose a general VANET aggregation model and finally discuss the additional attack types, introduced in Section 2.1.2, that play a role in VANET aggregation.

2.2.1 Motivation and challenges of VANET aggregation

To see the usefulness of VANET aggregation from the perspective of a vehicle, consider the resolution and quality of information sources at different distances, as shown in Figure 2.1. This figure shows a highway

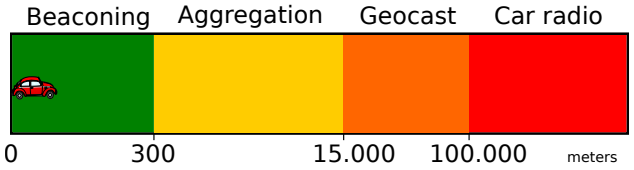


Figure 2.1: Different data sources with update frequencies at different ranges.

scenario and data sources that are currently available; each data source provides information from different locations, as shown on the x-axis. Color estimates the frequency and quality of the information, from high-frequency beaconing (Green) to low frequency radio broadcasts (Red). Newly introduced are the aggregation and geobroadcast areas, which provide medium-accuracy information over a relatively large area. Some aggregation only makes sense when performed over a limited area. For example, average speed or traffic density may be significantly different over a very long stretch of road; aggregating over an area that is too large may cause a traffic jam to be missed. Thus, between the aggregation area and the very low frequency of traffic information, the vehicle may obtain data that is forwarded, but no longer aggregated. Such forwarding should occur to some distance from the area to interested vehicles, which are headed for the area (eg. a traffic jam); the geobroadcast communication pattern is useful for such purposes [40].

In an urban scenario, traffic jams may be harder to detect because of traffic lights; it is difficult to define whether a traffic jam is occuring. However, for an urban scenario the example application of counting parking spaces is interesting. For this application, a figure similar to Figure 2.1 can be drawn, using a circle instead of a straight road and different ranges. Note that speed or density information, rather than traffic jam detection, may still be useful in an urban setting; for example, one could use this information to build traffic models and tune traffic lights for optimal throughput. However, the end-user application is out-of-scope for thesis; it focusses instead on the aggregation process and its security. The speed and traffic jam application will be used as a means to analyze schemes, as it is the most commonly mentioned application.

2.2.2 General VANET aggregation model

This section will provide an overview of how aggregation in VANETs occurs, providing a reference model on which different attacks can be explained. In previous work, Dietzel et al. [12, 13] showed how aggregation can be seen as a continuous process that stores the observations and received messages of a vehicle in a world model. The vehicle then selects interesting data for aggregation, places the aggregates in the model and forwards information to other vehicles.

For the communication between different vehicles, the model is shown in Figure 2.2. It consists of four roles, roughly representing the lifetime of an aggregate; observation, aggregation, finalization and forwarding. Each vehicle can perform one or more role; each role is seperate in the figure for clarity. The roles are split in two groups; an aggregation and a dissemination phase. The distinction between these phases is not explicit in current work, but it will be made explicit to use it in the new scheme that this thesis introduces. In the observation role (O), each vehicle obtains information from its own observations and broadcasts them to other vehicles in range. In the aggregation role (A), a vehicle combines received observations and aggregates, plus its own observations, to create one or more aggregates, which are broadcasted. At some point, a vehicle will decide the aggregate is complete (for example, when the average of speed observations stabalizes), creates a finalized message and forwards it (Fin), marking the end of the aggregation phase. This decision can be either a fixed threshold, or it could be based on the deviation of the aggregate from the observations of the vehicle that decides. In addition, note that finalization of a message can be represented either by a flipped bit in the message, or a more elaborate approach involving cryptographic signatures. Finally, in the forwarding phase (Fwd) this message is simply used and/or forwarded to interested vehicles; this is the dissemination phase. In the aggregation model for this thesis, which is motivated by a security background, it is not possible to 'de-finalize' the aggregate in order to further aggregate certain messages.

Note that in some schemes, the aggregation model will remain in the aggregation phase indefinitely,

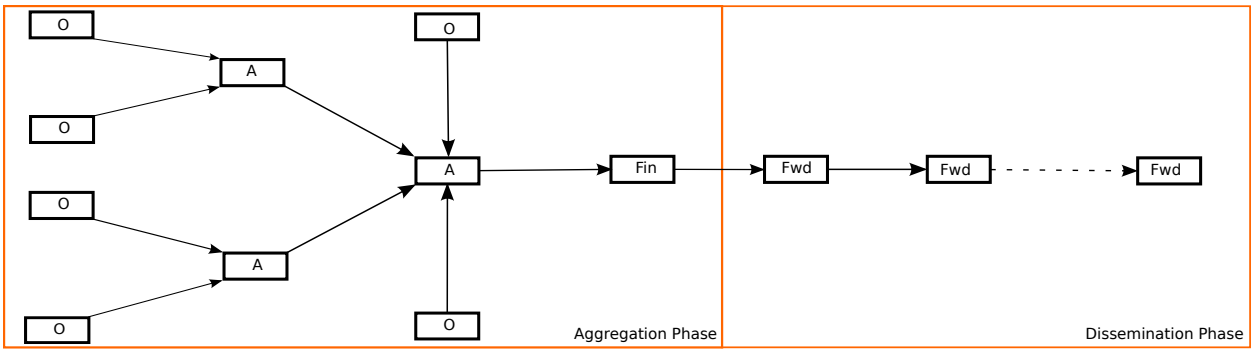


Figure 2.2: This figure shows the different roles that vehicles can have in VANET aggregation.

because they do not maintain a bound on the aggregation dimensions (ie. area and time), so no finalization will occur. These schemes are inherently vulnurable to attacks that make the aggregation area so large that the aggregate is no longer useful[5]. Another possibility is that the aggregate simply reaches an area where no new aggregation steps will be performed; the dissemination phase is entered implicitly.

In addition to these phases of aggregation, a core concept is that of order and duplicate insensitivity. This means that an aggregation process should give the same final result regardless of the order of processing and regardless of any duplicates that may be encountered during processing. Duplicates are here defined as distinct readings from different nodes[6]. An illustrative example would be determining the set of all nodes in some area: sets may only contain each node once ($\{A\} \cup \{A\} \equiv \{A\}$), and are insensitive to order ($\{A, B\} \equiv \{B, A\}$). This concept is inherently required for in-network aggregation; without duplicate insensitivity, either the aggregate will be subject to variation based on the paths in the network, or these paths must be fixed in some way to ensure exactly one copy of each reading is used in the overall aggregate. These solutions are commonly applied in sensor networks, but recall that for VANETs, neither of these solutions are satisfactory due to the network model. Therefore the authors of [34] have introduced a property called ODI-correctness, which will guarantee correctness of an aggregate produced by such an aggregation mechanism. Given that an aggregation method is ODI-correct, it can be applied in arbitrary network configurations, as long as it is suitable for the aggregation method and the data. Common examples of ODI-correct aggregation methods are summation and counting.

Fundementally, aggregation is limited in the information it can transfer; if one represents the aggregation area as a circle, then (in two-dimensional space, over which most applications aggregate) the message size[7] of the aggregate must not grow faster than $1/d^2$ [39]. For aggregation mechanisms, this growth speed is the optimal case, retaining as much information as possible without growing out of bounds to cause a broadcast storm. For additional security overhead, however, it is desireable to be as small as possible, while still providing confidence in the legitimacy of the message. If the size of the security overhead depends on the size of the aggregate, then there may be cases where security overhead would cause the scalability of a good aggregation scheme to be insufficient.

2.2.3 Security

Given the general VANET aggregation model, the attacks on aggregation discussed in Section 2.1.2 can now be distinguished using the model to identify the key areas where a secure VANET aggregation scheme should

[5]Although there may be solutions that mitigate this problem, such as secure positioning.

[6]Formally speaking, a duplicate occurs when an individual observation *value* has the same $\langle time, ID, area \rangle$, but in some cases this definition may be too wide. For example, if a node has a single sensor and two network cards, it may have two identities, but clearly two readings should be considered duplicates. Similarly, it makes sense intuitively to consider time and area in a fuzzier sense, limiting them to some granularity rather than exact values. Another issue would occur when two distinct values have the same tuple. These issues are not considered in this work; it is assumed that the tuple uniquely identifies the observation.

[7]This is for the entire message, including metadata and security overhead (if any).

employ a defense.

Consider the general aggregation model in Figure 2.2; each of the marked node types can be an attacker performing various attacks on the aggregation scheme. In this paragraph, it is assumed that normal messages in a VANET are protected against any modification, replay or injection, as achieved by current VANET security mechanisms [1]. Given that O is an attacker, she can attack the aggregation scheme by injecting false messages with her pseudonym; typically this happens in the context of a sybil attack. Given that A is an attacker, she may choose to perform a sybil attack by fulfilling the role of O many times, or she may choose to incorrectly aggregate the received observations and aggregates. This can happen in a number of ways, distinguished by whether A ignores unfavorable input, or an active bias is included in the aggregation scheme. Given that Fin is an attacker, she may finalize an aggregate right away, ending the aggregation phase early and causing a denial of service attack. Alternatively, she may inject a valid finalized message without ever having heard an aggregate; a remote impersonation attack. Given that Fwd is an attacker, she may perform denial of service attacks by dropping or jamming the packet, or she may attempt a replay attack by replacing the packet with an older packet. Note that the impact of such a replay attack can only be denial of service, as it is assumed that messages are protected against simple replays and injection by current VANET security mechanisms [1]. Since the messages are easily identified as old, the impact of this attack is very limited. Some of these attacks are very difficult or expensive to defend against, unless it becomes possible to link pseudonyms, which implies a loss of privacy.

Security can be considered as something that can be achieved in three ways, each of which provides trust from a different source, increasing the barriers an attacker has to overcome to perform a successful attack. The three distinct categories are cryptography, plausibility checks and interactive verification. These are VANET adaptations of the three possibilities discussed in [27]; cryptography, abnormality-based detection and retro-active detection. Cryptography can be used to provide trust by indicating how many vehicles were involved in the aggregate, to prove that a vehicle was indeed in the area specified by the aggregate and the integrity of the message itself. Plausibility checks are a useful tool to verify the correctness of a statement by simply checking it against models of physics, driver behaviour or simply by comparing with known statistics. This provides a confidence in a given aggregate. Finally, interactive verification refers to anything where two vehicles interact to verify certain statements. This paper will focus mainly on the cryptographic aspects for two reasons. First, current work does not provide a satisfactory solution; second, improving the guarantees offered by the cryptography component makes the other two more effective. For example, if a signature scheme that provides the number of signers is used, plausibility checks can use this as a parameter to compute the trust in a message. Furthermore, cryptography provides relatively conclusive evidence of a certain fact, while plausibility checks will always be probabilistic. Compared to interactive verification, cryptography may provide less overhead, depending on the amount of hops between the original sender(s) and the receiver.

2.3 Requirements

This section discusses the requirements for secure VANET aggregation. These requirements essentially summarize the main result of the preceding chapter.

2.3.1 Data utility requirements

Equal participation, meaning that every benign vehicle that has data available contributes to the resulting aggregate message in an approximately equal fashion, resulting in an aggregate that represents the 'average' of the data of all benign vehicles. For this definition, benign means any vehicle that is not controlled by the attacker **and** does not have malfunctioning sensors; therefore, vehicles with malfunctioning sensors can still be disregarded. Meeting this requirement also helps security, since if this requirement is met, then a higher number of participants implies that the receiver can have a higher confidence in the aggregate message. Conversely, it should not be possible for a single vehicle to have a disproportionate contribution to the aggregate; however, this definition is much closer to a security requirement.

Accuracy is the main requirement for the data structure or algorithm used to aggregate the data. Without sufficiently accurate mechanisms, secure VANET aggregation will not provide sufficiently useful information; in that case, applying secure VANET aggregation, or even VANET aggregation in general, is not sensible. In most cases, accuracy can be traded off against bandwidth efficiency.

2.3.2 Feasability requirements

Bandwidth efficiency is perhaps the most important requirement for secure VANET aggregation. One of the core advantages of using in-network aggregation as opposed to efficient dissemination and information gathering mechanisms is that the bandwidth consumption of in-network aggregation is much lower. However, aggregation reduces the value of the security payload that each vehicle generates; in efficient dissemination schemes, this security payload can still be used to verify the validity of the message. In aggregation mechanisms, on the other hand, messages are aggregated, which means the original security payload can no longer be used for verification. Secure VANET aggregation should not cause this advantage to be lost, since then security will be ignored, or aggregation as a whole will not be considered.

Computational efficiency here refers to the amount of time needed for cryptographic processing; it is thus required to provide up-to-date aggregates to interested vehicles. Insufficient computational efficiency will hamper the adoption of secure VANET aggregation mechanisms in favor of insecure ones, which will result in many security problems. Such security problems will effectively reduce an aggregation mechanism to a waste of resources, or even a risk to traffic efficiency. Computational efficiency may be achieved by hardware acceleration in some cases; in such cases the hardware used for this purpose should be sufficiently cheap to stimulate adoption. However, hardware prices are out of scope for this thesis.

2.3.3 Security requirements

Security requirements are the most important ones for SeDyA, as the goal is to provide security on top of the existing aggregation features that already exist. In this class of requirements, the most important ones are integrity for individual messages and data consistency for different aggregates. In addition, it is desirable to have privacy against other participating vehicles and a limited requirement on availability. Each of these requirements are shortly described here.

Message integrity refers to the integrity of individual messages in a single-hop broadcast scenario. This requirement is similar to that posed in general VANET security and the main purpose is to allow vehicles to detect message modification attacks. In addition, it provides the guarantee that all valid messages are sent by a vehicle, because the key material is certified by a certificate authority for use with a vehicle and key material is typically stored in a tamper-proof device.

Data consistency, on the other hand, refers to the intergity of aggregated messages. More precisely, the vehicle that performs the secure aggregation process should perform this task correctly. Therefore, the only thing an attacker can falsify is her own observation; she cannot abuse the merge process to generate false messages that are accepted because they are based on legitimate messages. Note that because aggregation fundementally modifies the content of the packets, it is not sufficient to require the to repeat the signatures attached to the observations it receives, because these signatures are generated on the observations, which are aggregated away, making them impossible to verify.

Similarly, **availability** should be achieved to at least the same level as achieved in regular VANET scenarios. In this context, availability refers to the absence of specific denial of service attacks, such as by means of injecting certain packets, or selective packet dropping, as opposed to general attacks such as simply jamming the entire communication channel. This is because general flooding-based denial of service attacks are practically impossible to protect against: even if the radio units are tightly controlled using hardware security, researchers have already developed an open source software-defined radio implementation of 802.11p, thus rendering such a control effort useless.

Chapter 3

Related Work

This section discusses the related work for the scheme in Chapter 4. In Sections 3.1 and 3.2 discuss probabilistic counting and cryptography, respectively. These sections, in particular that of probabilistic counting, are essential to understanding the issues involved in the construction of a secure aggregation scheme and the problems that current work does not solve. Section 3.3 discusses some current VANET aggregation schemes that do not consider security. The existing work that adresses secure VANET aggregation is discussed in Section 3.4; some of this work is actually intended for sensor networks, but may be adapted to operate in a VANET environment without breaking the ideas in the schemes.

3.1 Probabilistic Counting

A probabilistic counting algorithm is a method to count distinct elements in a set in a distributed system. These algorithms are also called distributed stream algorithms, which is a more general class that focusses on processing large amounts of data in a single pass in a distributed fashion. The original design goal for such algorithms was for databases to function in environments with little memory. This section will introduce FM sketches and some improvements suggested in [19], followed by a short introduction of LC sketches and the z-smallest method[1].

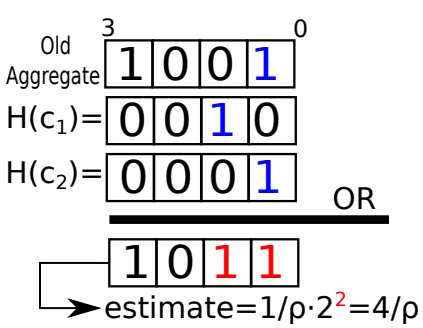
3.1.1 FM sketches

FM sketches are an instance of probabilistic counting, a method to provide smaller aggregates that can still be updated. They were originally developed for resource-constrained programs that processed large databases by Flajolet & Martin in [19]. In essence, FM sketches rely on counting hashes, as opposed to individual elements; because the same hash function is used by all nodes, FM sketches are duplicate and order insensitive. This is a tradeoff between transmission overhead and accuracy, when compared to transmitting all elements and counting them afterwards. Alternatively, if compared to a scheme that simply transmits the count, the FM sketches have less accuracy (in an ideal network), but offer both strict error bounds and order and duplicate insensitivity. The FM sketch is one of the most common distributed stream algorithms used for in-network aggregation schemes, in both sensor networks and VANETs [20, 24, 34].

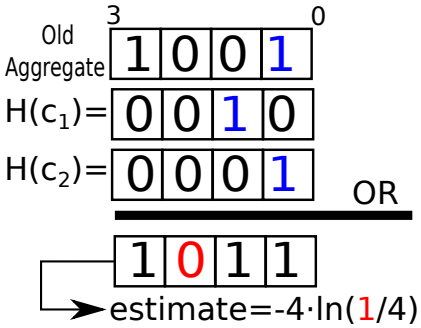
The operation of FM sketches is somewhat similar to Bloom filters in that it uses a hash function to map elements to a bit in a fixed length l bit string (a Bloom filter with its parameter set to 1). Given this, the FM sketch can then count up to 2^l distinct elements with a fixed error bound. The additional requirement for this result is that the mapping to the sketch is distributed geometrically. This mapping operation is typically implemented using a cryptographically secure hash function h, which provides a fixed output for the same input, but is otherwise assumed to be random[2]. Such a random hash function can implement a geometric

[1] Some notes on other candidate methods can be found in the research topics document

[2] This is the essence of what cryptographers call the Random Oracle Model. There is a lot more to designing secure hash functions, but this is not a requirement here.



(a)
Using FM sketches to count nodes c_i and add it to an old aggregate. Note the order of the bits is big endian.



(b) Using LC sketches to count nodes c_i and add it to an old aggregate. Because the hash function distributes uniformly over the bits, endianness is not relevant here.

Figure 3.1: FM and LC sketches

hash function as follows: compute $h(i)$ of item i and count the amount of zeros in the resulting bit string before the first 1 bit and use this amount as the output of the geometric hash function. To estimate the number of entries counted by a given sketch, the estimator $\frac{2^x}{\rho}$ is used, where x is the length of the sequence of 1 bits, starting from the least significant bit and $\rho \approx 0.775351$ [19]. An example is shown in Figure 3.1a; here, distinct c_i are counted and added to the old aggregate (which could for example be c_0), resulting in an estimate of $4/\rho \approx 5.16$. Note that it is possible to dynamically grow the size of the FM sketch, without invalidating old observations, as long as the last bit is not set. The reason is that the last bit is always set when it is reached, regardless of the result of the geometric distribution.

Using a probabilistic counting scheme like the FM sketch has a significant negative impact on the accuracy, due to the high variance in the estimate [19, 29]. To allow for a trade-off, probabilistic counting can obtain higher accuracy by implementing a technique called probabilistic counting with stochasic averaging (PCSA). Rather than using just one bit string (and associated hash function), multiple bit strings and hash functions are used and the estimation will be the average of the result of each: $\frac{m}{\rho} \cdot 2^{\Sigma_{j=1}^{m} x_j/m}$, where x_j is the end of the sequence of 1 bits in the jth sketch and m is the total amount of sketches used. [29] introduces a bound that is more accurate especially for sketches that contain less than $10 \cdot m$ elements; $\frac{m}{\rho} \cdot 2^{\Sigma_{j=1}^{m} x_j/m} - 2^{-\kappa \Sigma_{j=1}^{m} x/m}$, where $\kappa \approx 1.75$. Note that the hash functions should be distinct; this can be achieved by simply using $h_y(i) = h(i||y)$ as the yth hash function, where $||$ denotes concatination.

3.1.2 LC sketches

LC sketches are a variant of the FM sketches, designed to provide a higher accuracy and tighter error bounds [17] in exchange for a higher transmission overhead. The concept is the same as FM sketches, but instead of using a geometrically distributed hash function, a uniformly distributed hash function is used. The length of the sketch increases from $\log n$ to $m \leq n$ and the count returned after processing all the elements into the sketch is $-m \cdot ln(\frac{z}{m})$, where z is the amount of remaining 0-bits in the sketch. See Figure 3.1b for an example of an LC sketch. Accuracy of the sketch is given through the relative expected error: $\frac{(e^{n/m} - n/m - 1)}{2 \cdot n}$, given a set with n distinct items. Note that while LC sketches allow tuning of accuracy, doing so requires (some) knowledge of n. [17] claims a strong improvement of accuracy compared to FM sketches, given similar size; however, these experiments count sensor nodes directly. Countrary to FM sketches, there are no explicit claims about the usefulness of LC sketches for use with other aggregates like computing an average.

3.1.3 z-smallest

z-smallest is a probabilistic counting method that can be implemented in two ways. The difference between the two methods lies in whether the result can later be merged with another result, or whether it is purely used for probabilistic counting in one node. In the former case, the node counts a specific thing, such as neighbours it observes, and hashes each of them with $H \rightarrow [0, 1]$, which has a uniform distribution for its result. The node then simply keeps the z smallest values and transmits them in a packet, as opposed to transmitting all values at once. In the second case, the node only needs to perform the counting locally, so it can simply keep exclusively the zth element, selecting it as a representative for the entire set of values. This works because the hashes are uniformly distributed over a range, since given z and the corresponding hash value, one can estimate the amount of elements in $[0, 1]$ by simply computing z/v_z.

3.2 Cryptographic Signatures

This section will briefly introduce the basic principles of various types of signatures, each of which has a number of properties that are useful when designing a secure aggregation scheme. Throughout the past three decades, many researchers have designed cryptographic signatures for a variety of applications, including an aggregation setting. The signature types will be introduced and briefly discussed in the context of VANET aggregation. Because of the large scale and multi-hop setting, only signature schemes based on public key cryptography are used. Secret key cryptography, which is one of two options for SAS through the use of Message Authetnication Codes (MACs), does not support verification of a signature by an arbitrary vehicle and thus requires some scheme to switch to a new key and broadcast the old one for use. This is done in schemes like TESLA++ [45], but their use is limited to single-hop applications because of a broadcast storm problem, in addition to issues where due to retransmissions the key and message would arrive at a vehicle in the wrong order.

Signature A signature is a cryptographic primitive that provides proof that the sender of a message posesses a certain private key, because she is the only one that can create a valid signature that verifies using the corresponding public key. Thus, anyone can verify a signature, but only the owner of the key can create it. A valid signature only provides evidence that the signer is legitimate if her keys are not revoked by the CA and if the public key is obtained from a trusted source, such as a secure channel to the CA or a certificate containing a signature of the CA. For VANETs, the signature is usually sent together with the certificate, because it is impossible to pre-load all certificates of all vehicles. Recall that TBEV exploits the use of signatures to build a scheme that is akin to aggregation; she collects signatures from different vehicles on certain events.

Threshold signature [44] Given a group of n participants, each with their own private key (also called key shares), and a threshold $t < n$, a n, t-threshold signature scheme provides that a valid signature can only be generated when at least $t+1$ distinct keys are used. This signature will verify with the public key associated with the entire group of participants. In addition, an attacker that can compromise at most t participants cannot discover any information about the global secret key. Finally, some threshold signature schemes explicitly require that given a valid signature, an attacker can never discover which of the n participants participated in the signing process. Threshold signatures can be designed using a secret sharing scheme, such as [43] which generates a random polynomial f of degree t over $\mathbb{F}_m$, such that $f(0) = d$, where d is the secret, and computes the key share of each participant i as $f(i) \mod m$. Normally, to recreate the secret key, Lagrange interpolation can be used to recompute d. However, it is possible to allow each participant to generate its own part of the signature, after which these signature shares can be merged into a valid signature. In any case, threshold signatures require the message M to be fixed for all the signing participants. A valid threshold signature guarantees that at least $t + 1$ participants agreed to sign the message.

For VANET aggregation, threshold signatures are difficult to achieve, because the key shares need to be computed either beforehand or as a protocol is running. Neither solution is satisfactory, because the

amount of key shares is simply too great, even before pseudonyms are introduced. Alternatively, when the anonymity requirement is used, it becomes very difficult to detect sybil attacks. In such a situation, different messages can no longer be linked, as doing so would breach anonymity. In addition, the threshold t is fixed for the duration of its use, so it is impossible to shrink when the road is filled with too few vehicles (i.e., $\leq t$). Making t too low simply provides an expensive scheme that can easily be broken.

Multisignature Given a group of n participants, each with their own keypair (pk_i, sk_i) a multisignature on a message M can be generated by any subset of all participants L by computing it from the individual signature that each participant generates on M. Then, the signature can be verified by using all the public keys of the participants in L and computing from them the 'public key' of L. Multisignatures can be implemented in a discrete log setting by simply computing the product of all the signatures as the single signature; the public key is then computed as the product of the public keys of the participants in L. This scheme can also be applied to perform batch verification of signatures on the same message [6].

For VANET aggregation, this scheme can provide the same level of protection as threshold signatures, but in a more flexible fashion. Because multisignatures can be implemented using a regular signature scheme, without influencing the security of that scheme, its implementation into VANETs is also much simpler; there is no sharing protocol required to establish key shares. The most important drawback is that the set L must somehow be provided to the verifier of the message. Since the public key is already in most signed messages sent in a VANET, this is definitely possible. However, the amount of public keys will grow linearly with the amount of signers, which creates large amounts of overhead.

Aggregate signature [8] Given a group of n distinct participants and n distinct messages, an aggregate signature is a signature that can be computed from the signatures that are generated by each participant on its own message. Similar to multisignatures, each participant computes her own signature before aggregating them, and these signatures can be used normally. A verifier can verify the aggregate signature, given the signature and all n messages and public keys to convince himself that indeed each participant signed one of the messages. However, it is not necessarily possible for a verifier to determine which participant signed which message. Aggregate signatures may be implemented using pairing based cryptography, which will be discussed in Section 5.1.2.

For VANET aggregation, aggregate signatures seem to be ideal at first glance. However, notice that the requirement of distinct messages is strong for some applications, such as event validation, where the event is typically the same message for every vehicle, because sometimes the message of a vehicle may be identical to that of another vehicle. In addition, the messages sent with an aggregate signature are not aggregated themselves; instead, only the signatures are aggregated. However, as noted by SAS, aggregate signatures could be used to reduce the size of signatures that will not change, saving bandwidth at the cost of increased computational effort.

3.3 VANET Aggregation

This section briefly discusses a few important VANET aggregation schemes that do not consider security. The discussion given here is not intended to provide a complete overview of all available VANET aggregation mechanisms; instead it shows the evolution of different schemes and highlights the most important issues that have been encountered in these works.

SOTIS [50] is one of the earliest proposals of using VANETs to collect information about the surroundings of a vehicle, leading to the development of the typical local dynamic map (LDM, also known as neighbor table) that state of the art VANET protocols typically rely on. With a very low beacon rate (once per five seconds), SOTIS provides a coarse overview of the state of all the neighbors in the network. Aggregation is implicit and completely local; SOTIS only discusses that vehicles should compute data like the average speed of its surroundings and broadcast this data in its beacons.

TrafficView [11, 33], like SOTIS, is an information dissemination scheme that relates to the current VANET state of the art, using beaconing to gain knowledge about the surroundings of a vehicle. The work

also explicitly considers aggregation as a means to provide additional information to the system, providing two different algorithms to compute high quality aggregates. However, aggregation is performed only on transmission of the data, instead of when data is received. Instead, received data is first validated by a validation model, which makes the decisions on what to keep and which message is most up-to-date.

Lochert et al. [29] introduce the use of a variant of FM sketches to VANET aggregation. They argue that SOTIS and TrafficView cannot provide results, because they select an aggregate to store, rather than merging received aggregates. More fundementally, it is argued that this process of selection instead of merging bounds the best possible aggregates; the authors then propose the use of a variant of FM sketches, called the soft-state sketch, to avoid this issue. Soft-state sketches are FM sketches that use a counter instead of a bit; merging occurs by taking the maximum of all counters for each bit in the sketch. These counters each represent a time to live (TTL) field, which is decremented just before every transmission. Note that in this scheme, the sketches are not bounded to a particular region, lacking the ability to distinguish between areas where different values exist. For example, two traffic jams with an intermediate section where normal driving is possible would be aggregated into a single traffic jam.

TrafficMap [49] (see also the more advanced result in [41]) demonstrates the importance of dynamic aggregation areas when detecting traffic jams. It attempts to balance redundancy with accuracy by discussing an intelligent coding and filtering scheme to dictate which vehicles should aggregate when. This allows the scheme to closely match the actual situation on the road with a very low amount of packets and provide an accurate overview of a large section of the road, as well as be detailed enough for use as local information. However, unfortunately, the scheme the authors describe is specific to a specific type of data, namely average speed.

Dietzel et al. [12] discuss the use of fuzzy logic as one possible approach to obtain a more general system that allows for such accuracy. They take the ideas of TrafficView and replaces the knowledge database with a more detailed world model that represents a larger area of the network. Most notably, this database may contain conflicting information about the network, where fuzzy logic is used to decide which value is the correct value at a given time. Subsequent dissemination messages will use the fuzzy logic method to extract the current world view of the vehicle, which is then disseminated as the current state the vehicle has seen.

Both TrafficMap and Dietzel et al.'s fuzzy logic scheme show that dynamicness of the aggregation mechanism to match the dynamic reality of a road is important to designing a good aggregation scheme. Compared to their predecessors like SOTIS and TrafficView, they are able to disseminate information in higher quantities and quality. The probabilistic mechanism described by Lochert et al. shows that FM sketches can also be used in VANET aggregation.

Note that none of the discussed schemes consider security, although some of them use validation to check incoming messages and make decisions concerning them. The work by Dietzel et al. mainly considers a world model, so if the input to this world model can be provided in a secure way, and the dissemination mechanism is secure, then this work can make use of such security mechanisms. The other works show that a typical aggregation mechanism in VANETs has a high impact on integrity requirements (as also noted by Dietzel et al.); none of these mechanisms demonstrate that their scheme provides integrity. More frustratingly, their schemes cannot be extended with security in a straight-forward fashion (for example, by simply adding signatures), because this only provides the guarantee that the message was generated by another vehicle. The next section discusses some secure aggregation mechanisms that attempt to provide a stronger guarantee: that the aggregation process was performed in a legitmate manner.

3.4 Current Secure Aggregation

The idea to add security to aggregation in VANETs using additional cryptographic means was discussed first by Raya et al. [36]. As noted, this approach is fundamentally different from schemes that provide security through data-centric security mechanisms, such as [13]. Any solutions that are designed using cryptography provide suplementary evidence that can be used in addition to data-centric mechanisms, as cryptography can reduce the capabilities of the attacker by restricting the space of valid messages that she can use. Raya et al. introduced several options to secure an aggregation scheme. However, their ideas rely fundementally on

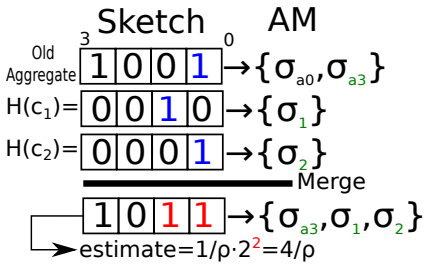


Figure 3.2: The FM sketch from Figure 3.1a with the AM for each sketch.

group communication, where all vehicle in a group are in communication range of each other. As noted, this requirement cannot be met by the scope of the current state of the art of the network model. It is assumed that such a cluster-like scheme would not be able to operate in a real world scenario, in particular in those scenarios where there is a slow traffic on one lane and fast traffic on another. The mechanism described in the paper is strongly dependent on this clustering mechanism, which is based on location cells; the group leader is elected by who is closest to the center. However, the group leader must be in posession of the public keys of the group members, and she must disseminate a group key to all of them. They propose three different signing methods to provide the necessary level of security; concatenated signatures (simply add all signatures in a list), onion signatures (sign the previous signature; provides only two valid signatures), and hybrid signatures (combining concatenated and onion signatures to have an arbitrary amount of valid signatures for half the overhead).

In current work, there are two approaches to provide secure aggregation in VANETs using cryptographic means; either adding additional cryptographic means, or by exploiting current mechanisms more effectively. Authentication Manifest FM sketches (AM-FM sketches) and SAS are both in the first category, where the main focus is inflation and deflation attacks [20, 24]. To prevent inflation attacks, both schemes apply signatures to specific bits in the FM sketch to protect its integrity across different messages, even after aggregation with similar messages. In the second category, Threshold Based Event Validation (TBEV) uses a fixed message format to which all protocol messages must conform. With this decision and the introduction of a new protocol, TBEV provides secure aggregation without introducing additional security. Both AM-FM sketches and TBEV limit the scope of aggregation mechanisms to binary events, such as a warning of a traffic jam or a collision warning. For AM-FM sketches, an alternative method is discussed to still be able to compute (approximate) sum and average queries with binary events. TBEV does not consider such aggegation use cases, but a similar approach may be used, as long as events are defined in advance.

Finally, note that current work considers sybil attacks to be impossible or very limited due to the use of at most one pseudonym at any time, or by completely avoiding them by not applying any pseudonym scheme, thus ignoring privacy issues.

3.4.1 AM-FM sketches

AM-FM sketches are a scheme designed for originally sensor networks, which can also been applied to VANETs. The fundemental idea is to generate some proof in the form of signatures on observations that can be transferred to validate the aggregate, indicating that the aggregation procedure has been executed properly. To keep signatures verifiable without using complex and expensive cryptographic primitives, each node generates a signature for every 1-bit in the FM sketch. Each node has a unique identity and signs its tuple, where a tuple is the sensor value for a binary event. Attached to each 1-bit in the FM sketch is the position of the bit, a unique identity, a signature and a tuple that resulted in this bit being set. The example FM sketch from Figure 3.1a is extended with an AM in Figure 3.2, which also shows how to merge the AMs of different messages.The merge operation only keeps one signature per bit, to save signatures. This example only counts nodes that observe a binary event; however, it is possible to extend AM-FM sketches to compute sums. In the original paper [20] an iteratively queried list of binary events is used, but this will not

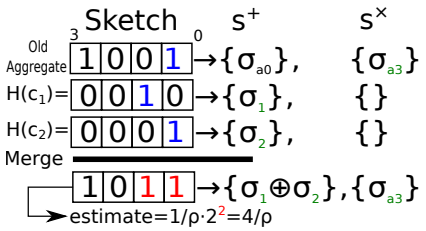


Figure 3.3: The FM sketch from Figure 3.1a with the inflation-free and supplement proof for each sketch.

be practical in most VANET scenarios. Given this, it is possible to either discard the iterative component, introducing a significantly larger error bound, or to adapt the method used to extend a normal FM sketch for measuring sums, an approach used by SAS (see the next section), which sacrifices some security.

Finalization and forwarding phases are not implemented in this scheme, as its intended application is sensor networks where aggregation is on-going until the aggregate reaches the querying node. As noted in the discussion in [20], the AM can be translated to other probabilistic counting algorithms; as long as it is possible to associate an observation with a signature that remains verifiable and these signatures can be collected in a compact set.

Security AM-FM sketches are intended to protect against in- and deflation attacks; it is assumed that sybil attacks do not apply because of a unique key, registered at the sink node. Remote impersonation attacks do not play a role in sensor networks. Inflation protection is achieved by the signatures; the attacker must compromise a key to flip arbitrary bits to 1-bits in the sketch. Flipping arbitrary bits that are not supposed to be 1-bits will reveal the attack because of a missing signature; if the attacker uses her key to sign the appropriate bit, a significant inflation would require the attacker to sign multiple bits, revealing her identity and the attack. In the case of a sum computation using the iterative component, an attacker would still appear multiple times in different predicate polls if she attempts to inflate the value, resulting in a revealed identity and revealed attack.

Deflation prevention is not natively achieved by using AMs; the authors in [20] propose to also include a response for the inverse query, such as to obtain a complementary FM sketch. The issue that arises from this method is that a sink node that obtains the sketch for verification must know the amount of nodes that participate. This problem is not solved by the authors, who argue that in most sensor networking applications a good estimate of this amount is known. The authors also note that an approach with weaker guarantees about the bounds of the aggregate would be to exploit redundancy in the network. Finally, in the case of more complex queries like a sum or average, inverting a query may be more difficult; however, for the iterative computation method described in [20], the inverse queries are given.

Overhead Recall that typical probabilistic counting algorithms, as discussed in Section 3.1, are query-based rather than continuous, so using query identifiers for the signatures does not introduce additional overhead. However, a query will need to be generated and disseminated to the sensors. The query can be protected by a signature, preventing modification of the query identifier before the query reaches the node. The messages sent by each aggregating node will contain an AM, which in turn will contain at most $k \cdot \log n$ signatures, where k is the amount of sketches and n is the maximum length of the data. Note that n is the range of the data type multiplied by the amount of participants, because of the way FM sketches work with non-binary queries (see Section 3.1). In addition to these signatures, the FM sketches themselves should be included along with a typical signature to ensure the integrity of the message as a whole. In the sensor network setting of the original paper, it is reasonable to assume that the querier knows all the public keys of her sensors. However, in a VANET setting, all the public keys need to be included to ensure that the signatures can actually be checked, creating additional overhead.

3.4.2 SAS

Another proposed solution that uses FM sketches as a basis is SAS. SAS eliminates the need for inverse queries as used in AM by using a different proof strategy, consisting of three components. These components are the inflation-free proof, deflation-free proof and supplement security proof; given these proofs for each observation, an aggregator can compute a new three-component proof for the aggregate, after performed aggregation. These proofs consist of specific information about the data along with a number of message authentication codes (MACs), which perform a task similar to the signatures in an AM-FM sketch. The MACs are generated using a secret key, which is shared between a vehicle and a central authority, the traffic monitoring center (TMC). This TMC is interested in continuous data from pre-determined segments of the road for analysis. In [24], the authors mention that SAS may also be used with asymmetric cryptography, replacing MACs with homomorphic signatures. However, in the paper itself this is only mentioned at a few points and not thoroughly analyzed, leading to the assumption that the version using MACs is the most relevant one. This especially because the use of homomorphic signatures brings with it an additional computational cost and further privacy issues.

The inflation-free proof contains a MAC for each 1-bit in the sequence before the first 0-bit in the FM sketch, along with epoch and location identifiers. The supplement security proof contains MACs on the remaining 1-bits and the epoch and location identifiers. Thus, each 1-bit in the FM sketch is authenticated at any time; when two sketches are merged, at least one MAC authenticates any bit, since there are MACs for every bit in both sketches. This approach is similar to that of AM-FM sketches; Figure 3.1a is repeated with the appropriate proofs in Figure 3.3. While it is not made explicit, each MAC must also be linked to a global identifier for the user, so that the TMC can determine which key it should use to check the MAC. Finally, the deflation-free proof is a MAC on just the epoch and location identifiers, which is used as the seed of a hash chain. A hash chain is formed by repeated application of a one-way function F; the root of a hash chain is the input of the first hash operation. The xth element of this chain is then denoted $F^x(y)$ for input y. Each vehicle sends not the MAC of the deflation-free proof, but $F^k(s_i^-)$, where s_i^- is the MAC of the deflation-free proof and k is the position of the first 0-bit of the sketch. For the merging procedure, all the aggregator has to do is make the chains equally long by applying F exactly as often as the difference between the 0-bit position in the original message and the aggregate. This difference will always be non-negative, because 1-bits can never become 0-bits in an FM-sketch, so the aggregator will never have to reverse an application of F. Because the TMC knows the location, epoch and key, it can simply compute the deflation-free proof and use it to recompute each hash chain. In the actual implementation, an additional operation is introduced to fold all the hash chains into a single one, so only a single hash is required to be sent in the actual message.

Security Since the application scenario is specific to traffic information, inflation and deflation attacks are the most interesting attacks. Sybil attacks are not treated in the paper; this is reasonable, because there is a single shared key between a vehicle and the TMC. Remote impersonation attacks are not mentioned; however, such an attack would be quite easy to detect, as no other vehicle will aggregate its messages with the message of the attacker and she can be identified directly from the message.

The functionality provided for collecting data is better than AM; notably SAS is not bounded to count only binary events. However, the approach sacrifices some security; even though each 1-bit in the FM sketch is signed, it is possible for the attacker to act as an aggregator and compute a fictious observation that sets exactly the desired bits in the aggregated sketch. Detecting this attack is very difficult, because unlike binary events it is very possible that a user's identity will occur multiple times in a single sketch. In addition, SAS requires that the aggregator always preserve the signature of the lowest identity for the inflation-free proof. This is not strictly necessary, but it is impossible to use a random signature like AM does, because the procedure needs to be reproduced by the TMC, because the MACs are XOR-ed to preserve space. It may be possible to detect the attacker if she sets a disproportionate amount of bits; however, typically the attacker needs to set only a few bits to significantly influence the aggregate.

Concerning deflation-freeness, the attacker can attempt to just flip 1-bits in the aggregate and try to discard the associated MACs to perform deflation. As an aggregator, the attacker can do this by refusing

to set a 1-bit in the aggregate and discarding the associated MAC from the supplementary security proof. However, due to the hash chain in the deflation-free proof, the attacker needs to reverse the hash function to be able to compute the correct folded hash, which is hard. The messages that are sent to perform aggregation should still be signed, in order to avoid issues where an attacker changes the identity in the inflation-free proof.

Overhead The total amount of MACs correlates directly with the amount of 1-bits k in the FM sketch: there are a total of $k+1$ MACs in a single message. To solve this issue, the original paper introduces a simple optimization; the MACs in the inflation-free proof can be XOR-ed, since their individual knowledge is no longer required. It is claimed that the sketch proofs all together produce an overhead of $8 \cdot log_2(v_{max} \cdot n) + 136$ bytes, where n is the amount of vehicles in any section and v_{max} is the maximum speed. However, this claim is largely not motivated, except for the fact that a sketch will indeed contain $log_2(v_{max} \cdot n)$ bits. Given the optimization mentioned earlier, the total security overhead should only be the size of $1 + y$ MACs (where y is the amount of bits signed in the supplement security proof), plus a single result of the one-way function used for the deflation-freeness. For the latter, the paper proposes the use of 1024 bit RSA, producing an additional 128 bytes of overhead. Assuming a MAC is 8 bytes in size, this corresponds roughly to the computation provided above as a worst-case scenario.

Disadvantages Every bit of privacy the driver had towards any receiver of the messages is lost; the introduction of privacy into SAS would break both the security and efficiency of the scheme. The identity of the vehicle must be known, because it is necessary to identify the correct symmetric key with which the TMC should check the signature. Note that the introduction of privacy would require either a regular change of symmetric keys for every vehicle, allowing sybil attacks, or some kind of anonymity guarantee that could be provided by an aggregate signature scheme. An aggregate signature scheme is indeed mentioned as an alternative to using common MACs; however, its efficiency in both size and space is not evaluated. Even with such a scheme, privacy towards the TMC will be difficult to obtain without opening the way for sybil attacks.

Another problem is that since the identity is never included in any signature, it is trivial to perform a denial of service attack on the entire block (location and epoch identifiers) by simply sending a single mismatching identifier with a key. This attack can be performed by any vehicle, as long as it is a participant in the protocol.

Finally, the paper does not address communication delays or computational efficiency in its analysis of the protocol. Rather, a graph of the standard error and a graph showing the overhead for different amounts of sketches are given. From these graphs, it can be seen that to obtain a reasonable standard error the use of at least 8 FM sketches is required, regardless of the use of SAS.

3.4.3 Threshold-based event validation

The third scheme takes a different approach; instead of securing any arbitrary aggregate, [25] focusses on protecting against what the authors call decision changing attacks on binary events. These attacks include any type of attack, as long as it can change the aggregate in such a way that a decision based on the aggregate changes (so remote impersonation is not considered here, because such an attack introduces a completely new aggregate). However, it is assumed that attackers cannot obtain many keys within the same time of validity to execute sybil attacks. This approach significantly simplifies the problem, leaving some applications out of the picture, but it allows the construction of a secure scheme.

To achieve protection against decision changing attacks, standard signatures are applied to protect individual messages, and a specific message format is enforce for aggregate messages. Rather than applying additional cryptography, this enforced message format improves the effect of standard signatures. Vehicles are able to send a fixed number of events; given an observation of event ξ and a coarse-grained indication of time and location (estimated to be 10 minutes and the nearest intersection or highway exit), TBEV provides a protocol to efficiently disseminate the event over multi-hop. This protocol, the Message Exchange Protocol (MEP), does not contain additional security guarantees beyond simple blacklisting to avoid denial of service

attacks, and thus is not relevant to this discussion. What remains then is to count the different reports of an event ξ and use threshold τ to determine whether the event is accepted as having occurred. The scheme consists of a total of four parameters τ, a, b, δ which represent the threshold τ, the noise zone $[a, b)$ around this threshold and the bound δ on the false positive and false negative rate, respectively. If the result of counting occurances of an event lies outside the noise zone, then with high probability this event indeed occurred.

Security The security of this scheme is determined by the well-known properties of signatures that are expected to be used in normal VANET applications. In fact, MEP specifies piggybacking a message digest on each beacon. Because the actual protocol only allows the transmission of a fixed range of events together with the signatures and certificates of the vehicles that have encountered this event. This approach avoids issues with complex compression of signatures or providing proofs that still verify after aggregation, but limits the applicability of the scheme. On the other hand, it shows that cryptographic signatures can be effective in an aggregation setting. The only security concern is weighing the privacy of the drivers against the possibility of sybil attacks. The authors argue that privacy is protected by the use of pseudonyms, but also assume that only one pseudonym is active at any time, which has been observed as a very strong assumption in previous work [23].

Overhead The overhead required by this scheme is bounded by the probabilistic counting mechanism that is used to count the events. This not because the probabilistic data structure is transmitted, but because it provides a bound on how many signatures should be distributed per event as part of the MEP. Because of this, FM sketches are not the ideal choice; instead, z-smallest is used as a local scheme, which means only one value needs to be kept. For overhead, this then means the size of the synopsis is bounded by $O(\frac{ln(1/\delta)}{\epsilon^2})$, where $\epsilon \in [0, 1]$ indicates the lower and upper bounds of the estimate, which are $n \cdot (1 - \epsilon)$ and $n \cdot (1 + \epsilon)$ respectively. As a practical example, [25] indicates a synopsis size of 128 elements for when the amount of vehicles $n = 10000$, $\epsilon = 0.1$ and the false positive/negative bound is 0.05. Each element in the synopsis is a signature and a certificate, providing a total element size of 181 bytes. The synopsis also includes an event description, though it is only 136 bits (17 bytes); this synopsis is attached to every beacon to allow the vehicles to detect that their neighbours have new events available.

A concern of importance is the amount of possible events. In order to use this scheme for a general aggregation application, such as average speed measurements, many different events need to be composed, causing the event space to grow as well as the variety of synopses. For fairly accurate measurements, for example at a granularity of 5 km/h, it is desirable to merge two different events (eg. $50 - 55$ km/h and $55 - 60$ km/h). However, this is not possible in TBEV. In addition, a large variety of events means that more synopses need to be kept, and thus more signatures will be broadcast.

Chapter 4

SeDyA: overview

This chapter will introduce the conceptual ideas behind the main result of this thesis, a scheme called Secure Dynamic Aggregation (SeDyA). This chapter will focus on effective aggregation mechanisms and appropriate use of cryptography, without going into details. Cryptography will be treated as black boxes in this chapter; their discussion is deferred to Chapter 5, which will explain what kind of cryptography is used in SeDyA. SeDyA is inspired by ideas from the related secure aggregation schemes discussed in Chapter 3, but it will address two main shortcomings; in the current secure schemes, aggregation is static and the security guarantees are insufficient to provide the desired level of protection. In addition, because aggregation is not dynamic in related secure aggregation schemes, long-distance dissemination may have bandwidth overhead issues. However, solving these issues is not the main goal of SeDyA; it instead aims to provide a trade-off between security and bandwidth consumption that provides a higher level of security. The remainder of this chapter is organized as follows: A high-level overview of SeDyA is given in Section 4.1, including the motivation for the division of SeDyA into three phases. After the overview, each of the three phases are discussed in detail. In the discussion of each phase, the main issues are briefly explained along with the solution that SeDyA provides.

4.1 Overview

SeDyA consists of three phases; the aggregation phase, the finalization phase, and the dissemination phase. These phases relate directly to the aggregation model from Section 2.2.2; the model itself is repeated in Figure 4.1. This model can be mapped to a real world scenario with three different areas, in which the phases perform their role; the areas are shown in Figure 4.2. Each vehicle that plays the role of observer or aggregator (O or A in Figure 4.1) is in the aggregation phase, attempting to find a locally stable value.

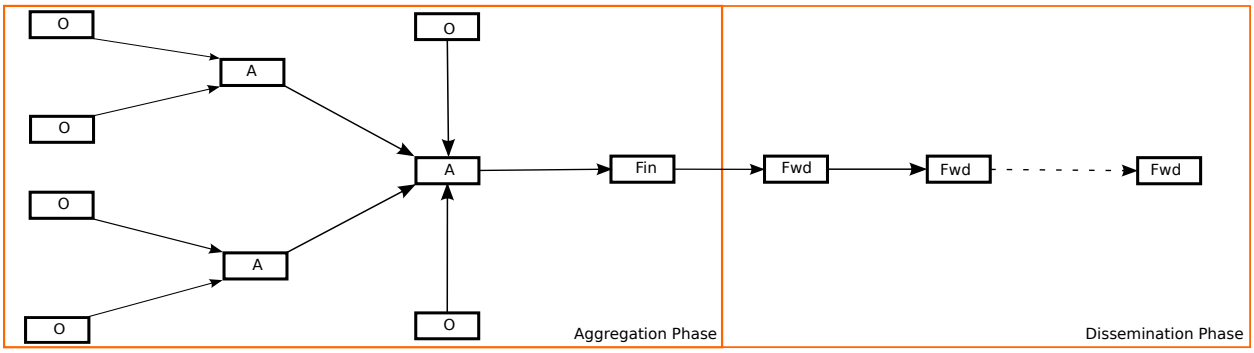


Figure 4.1: The aggregation model introduced in Chapter 2.

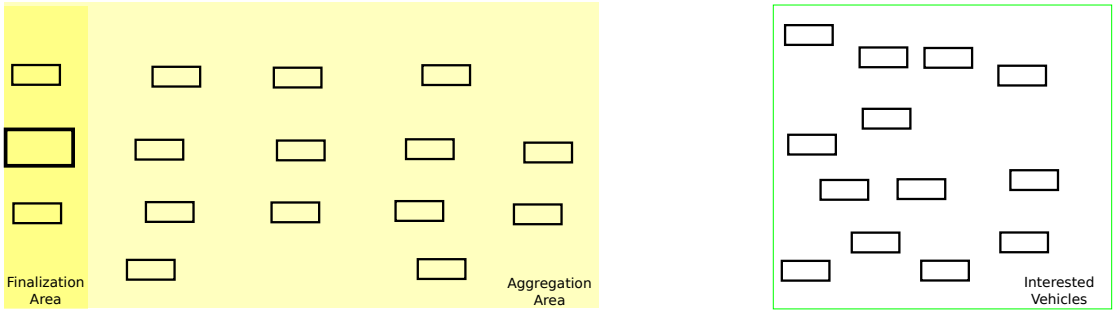


Figure 4.2: The different regions in SeDyA. Each phase will operate on a different subset of this figure; this figure will be repeated for each phase, marking the elements that are significant in that phase.

When such a value is found (by one or more edge nodes, one of which becomes a finalizing vehicle, denoted Fin in Figure 4.1), the finalization phase begins, where some vehicles finalize the aggregate and forward it through the aggregation area. This is the point where SeDyA deviates from the model; it first communicates the message through the aggregation area. As the aggregate is forwarded through the aggregation area, each contributing vehicle has the opportunity to sign the aggregate. The finalization phase is where SeDyA's security supersedes that of the related secure aggregation schemes. Once the aggregate has been passed through the area and vehicles have chosen to sign or not sign the content, the aggregate implicitly enters the dissemination phase (denoted by the vehicles labeled Fwd in Figure 4.1), where the aggregate is only forwarded to inform as many vehicles as possible. SeDyA provides additional security through security mechanisms in each of these phases, the ideas of which will now be presented.

Aggregation phase The purpose of the aggregation phase is to find locally stable information, such as the average speed, which can then be disseminated to the interested vehicles outside the aggregation area. To perform this task, AM-FM sketches are used in conjunction with AM-LC sketches. AM-LC sketches are an extension of LC sketches, applying the same mechanism as AM-FM sketches to protect against attacks, as discussed in Section 3.4.1. In addition, the aggregation phase applies these datastructures to describe the aggregation area in a dynamic fashion, as opposed to related secure aggregation schemes. These related schemes all select a short, fixed stretch of road as the aggregation area; aggregation cannot occur between two messages from different areas. SeDyA will allow discrete events, such as an average speed[1], to be described, as opposed to the binary events that are used in some related secure aggregation schemes. However, the security of this type of scheme is not as strong as when binary events are used (like e.g., in TBEV); the attacker may be able to make subtle changes to the aggregate to produce a different result with very little risk of detection.

Finalization phase To provide the stronger security guarantees that SeDyA aims to provide, the finalization phase is introduced. The first step in this phase is to determine when to consider locally stable information as final and in which area the information is stable. Once this choice is made, the message (i.e., the aggregate) is finalized by creating a multisignature on it, indicating it as a finalized aggregate that describes the area indicated in the message. A multisignature, in its simplest form, is the product of multiple signatures on a single message, such that the resulting signature can be verified with the product of the involved public keys. Using multisignatures as opposed to normal signatures provides a constant size

[1]Technically, this is a continuous event, but it should be made discrete to be compatible with sketches. Note that this will not add significant error compared to the error produced by sketches, as previously discussed in Section 3.1.

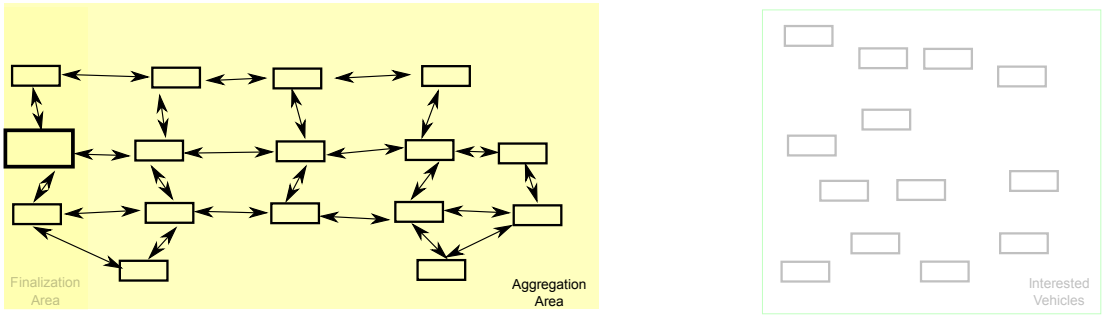


Figure 4.3: The areas of Figure 4.2 relevant to the aggregation phase.

signature and significantly reduces verification time, although this is still $O(n)$ in participants[2]. To compute the multisignature in a distributed fashion, a protocol is introduced, which forwards the message through the aggregation area and allowing each vehicle to add its signature to the multisignature. The security payload from the aggregation phase is dropped after this phase.

Dissemination phase Finally, the message will be disseminated from the aggregation area to the interested vehicles in the dissemination phase. Dissemination will occur using any standard broadcast dissemination scheme; the main concern for SeDyA is the guarantees it can provide, how attackers may be detected en-route and relating to this, how to detect and minimize the threat from attacks if the aggregate arrives at an interested vehicle. These tasks are mainly addressed by using heuristics and an existing world model, where confidence values are assigned to each piece of information. The heuristics indicate how SeDyA's guarantees and attack detection may be quantified to provide confidence values for the information that is transmitted. This allows SeDyA to be used in the related research field of data consistency, where the vehicles need a variety of sources to determine what information to use; SeDyA will be one of these sources.

4.2 Phase 1: Aggregation Phase

The aggregation phase is mainly concerned with finding locally stable values that are of potential interest to vehicles at a larger distance. The type of information considered includes current traffic density, available parking spaces in the vicinity, current road status, weather conditions and average speed. For example, most vehicles on a highway will be interested in the traffic density in the subsequent 10 to 20km ahead of them, to determine whether a traffic jam is likely to occur, and whether it makes sense to take the next exit to circumvent it. The relevant vehicles in the aggregation phase are indicated in Figure 4.3; only those vehicles that are in the aggregation area will be considered.

For the aggregation phase, there are three important issues: selecting the aggregation area, providing the maximum possible accuracy and limiting the overhead in terms of bandwidth usage. These issues relate directly to the discussion of the aggregation model (Section 2.2.2) and the network model (Section 2.1.1).

The main goal for the attacker in this phase is significantly influencing the value that the aggregate provides, which allows the attacker to use the benign vehicles as a stepping stone to convince the interested vehicles of a particular state of the road. The stepping stone effect is caused by the fact that many vehicles are involved in the aggregation process, and the receiver of the aggregation message can only distinguish the inputs of different vehicles by the signature on each bit. Recall that AM-FM sketches are used for security,

[2]Verification of a multisignature requires a single signature verification and n group operations to compute the product of the public keys. Typically, this operation takes much less time than a complete signature verification.

so each bit is signed; however, since SeDyA allows discrete inputs, a single vehicle is allowed to set and sign more than one bit (unlike the AM-FM approach, which allows only binary inputs). The only possible way to detect an attack with this information is by over-representation of a particular signer; the attacker can avoid this by adaptively selecting the bits she flips. This reduces the impact somewhat, but will make it practically impossible to distinguish the attack from a legitimate contribution. Such an attack is an in/deflation attack: this type of attack is the most important one in the aggregation phase. On the other hand, sybil attacks and remote impersonation attacks are relatively challenging to perform, because the aggregation phase is a relatively short-lived phase. For privacy purposes it is assumed that pseudonyms will be used, but that each of these will be valid for a short period, with limited overlap, making it more difficult for the attacker to perform sybil attacks. This is similar to what related work requires as protection against sybil attacks.

4.2.1 Dynamic aggregation area

The effectiveness of the aggregation phase strongly depends on how the aggregation area is chosen. There are two fundamentally distinct methods to choose the aggregation area; either by a predetermined interval, or dynamically based on the deviation from other atomic observations and neighborhood information. Although it is desirable to have a dynamic aggregation area, providing security for such an area is very challenging. Recall from Section 2.2.2 that aggregation with support for many types of information requires identifiers to avoid mixing different aggregates together unintentionally. Because in-network aggregation is distributed and does not rely on a querying node, there is a need for using part of the input for aggregation as information that uniquely identifies an aggregate, so that unrelated aggregates can be distinguished from each other. Most commonly, the aggregation area is used, which typically consists of two or three spatial dimensions. However, these dimensions are not known to every vehicle before aggregation occurs. Therefore, existing schemes resort to introducing granularity to enforce a particular structure over which aggregation can be performed. This identifier is then used as a fixed piece of information that may be cryptographically signed to provide authenticity, a property that is exploited by both the SAS and AM-FM schemes that were discussed in Section 3.4. TBEV also uses a granularity, but it is more restricted than those of SAS and AM-FM, because the granularity is very coarse, allowing TBEV to provide sufficient security at the cost of less accurate data.

Certain aggregation schemes that do not consider security offer a dynamic aggregation area instead of a fixed one. However, most of these schemes intelligently merge different aggregation areas to larger aggregation areas, which makes securing them with cryptographic signatures very challenging. In general, any security mechanism for such a scheme would need to allow other nodes to modify the data in the message, while still providing the expected guarantees on verification of a signature. Note that providing the ability to modify cryptographically signed data can be achieved already, by exploiting homomorphic properties. However, the challenging part is to cope with insider attackers that abuse such a homomorphic property. For this reason, SeDyA builds on the related secure aggregation schemes, rather than attempting to design security around the aggregation schemes without security. The AM-FM approach is modified to support a dynamic area that is indicated by several sketches. The security mechanism remains the same as that of regular AM-FM sketches.

The aggregation area can be dynamically selected based on several fixed points. As noted, such specific points must be predetermined and consistently selected, because they are used as input for the signing procedures in the sketch and should thus be the same for all vehicles in the aggregation area. In a highway scenario, the road on which an aggregate is computed, is typically a single region where most vehicles drive in the same direction. Thus, an aggregation area in a highway scenario will only vary in one dimension. This can be exploited by computing the average of all locations of the participating vehicles, as shown in Figure 4.4. The vehicles in the finalization area will indicate their location in the finalization phase, specifying one edge of the area and allowing receivers of the aggregate to reconstruct the area by symmetry in the average. Thus, the area is defined by the average and the location of the finalizing vehicle; the other end of the aggregation area can be computed by symmetry of the location in the average location. This is based on the assumption that vehicles with similar sensor readings (such as speed or road conditions) will strongly correlate with a similar traffic density. However, in an urban scenario, aggregation will typically occur across a network of roads. Depending on the scenario, it makes more sense to aggregate over the whole area, or

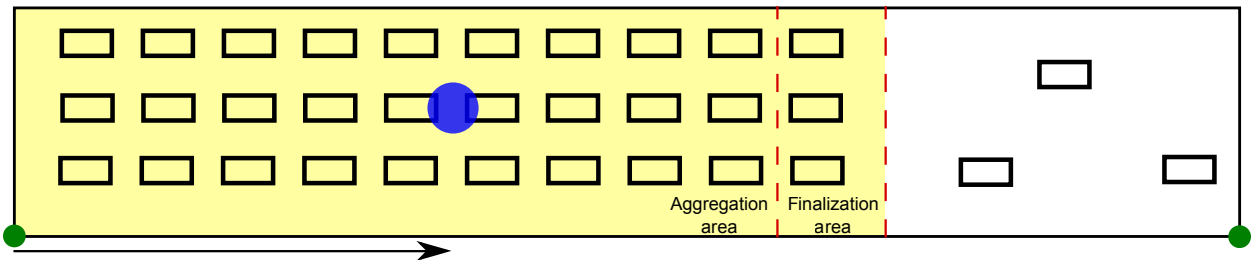


Figure 4.4: This figure shows a single direction of a highway with a traffic jam in the aggregation area. The left- and rightmost dots indicate fixed map points, while the dot in the center indicates the average of the distances. This information is stored in the sketch as a single vector, shown below the road.

to consider individual roads separately. Aggregation over the whole area is sensible only when the roads are comparable; if an urban road network consists of a large main road with mostly empty side-roads, then aggregation over the whole area will only add inaccuracy. This issue is shown in Figure 4.5, where a traffic jam on one three out of four edges in a crossing cause the receiver of an aggregate to see the empty section as part of the traffic jam. Therefore, if it is known that the road on which a vehicle is driving is the only road with high-volume traffic in the area, SeDyA can specify that this road should be considered a highway.

4.2.2 Accuracy

A second important issue is the accuracy of the aggregate, which directly relates to the issue of message overhead; higher accuracy will fundementally require more bandwidth. Ideally, if there were no bandwidth requirements, all messages could simply be forwarded to all interested vehicles, providing 'perfect' accuracy. Because aggregation is performed, less bandwidth will be used, but thus it also becomes important to consider accuracy. Inherently, aggregation transmits less information and thus the data structure will have inaccuracies compared to transmitting all data. More concretely, secure aggregation schemes from related work use either a coarse granularity to limit the possible messages or a probabilistic counting technique with a small data structure like an FM sketch. In either case, the data is encoded into the specific structure to save bandwidth, as well as to obtain the ODI-correctness property. See Section 3.1 for more details on probabilistic counting techniques. For probabilistic counting techniques, accuracy can be estimated by analyzing the error bounds or the relative error even when they are used without considering network conditions. The relative error will be used in a preliminary analysis in this section, to choose how probabilistic counting should be used in SeDyA.

The chosen probabilistic counting settings can then be used to create a secure scheme that finds locally stable values in a dynamically chosen area. The choice for this settings is a choice between the size and type of sketches. For the latter, there are two options; FM sketches and LC sketches, both of which have the useful property that a particular bit in the sketch will not flip to zero after it was set to one. This property makes possible the security mechanisms that the related secure aggregation schemes AM-FM and SAS use. Both of these approaches sign the position of each one bit, together with static information. The AM-FM approach can trivially extend LC sketches, obtaining AM-LC sketches. However, the AM-FM approach has two distinct disadvantages: the static aggregation area and the requirement of using an interactive approach to compute sums. While the static aggregation area has already been addressed in the previous section, the interactive mechanism for computing sums remains. This issue can be addressed by introducing the same mechanism for computing sums as already existing in SAS and regular FM sketches, as discussed in Chapter 3. Simply put, this approach requires vehicles to add k values to the sketch, where k is the value that the vehicle is adding to the sketch. For example, a measured speed of 10 kilometers per hour for vehicle 0 would be added by inserting $(0,1),(0,2),(0,3),...,(0,10)$ into the sketch. However, this introduces inaccuracies, as noted by [20], which cites it as the reason for introducing interactivity. In VANETs, however, interactivity is not an option: SAS simply uses the sum computation approach, and so will SeDyA. This also

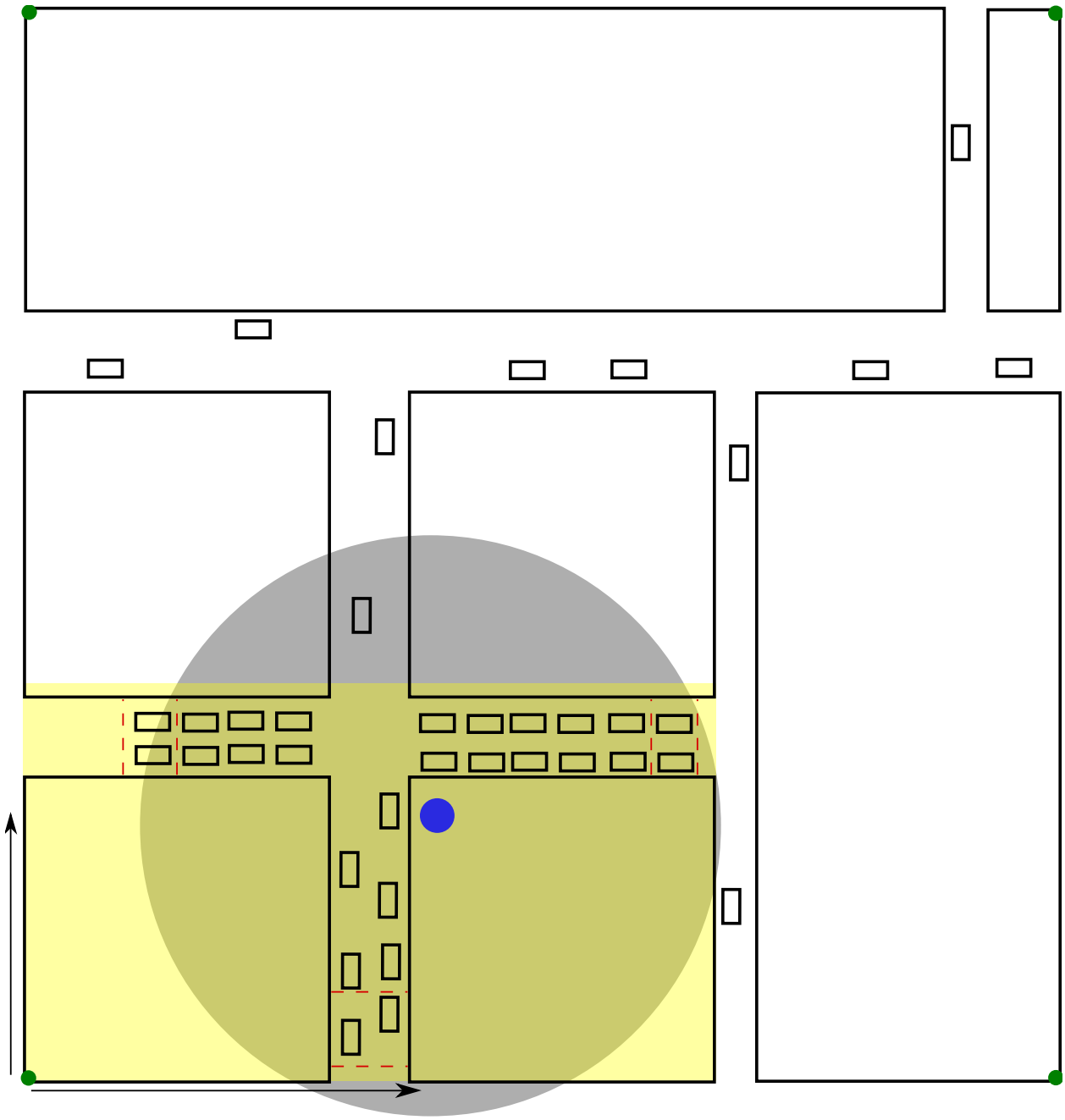


Figure 4.5: This figure shows a typical urban scenario with seven buildings and a traffic jam in the bottom left region. This figure has three finalization areas, on the outer ends of the traffic jam. Again, the corners of the figure have the fixed points from which sketches count. The dot approximately in the center indicates the average location of the vehicles in the rectangle that indicates the vehicles involved in the aggregation process. When a vehicle receives a finalized aggregate from a finalizing vehicle in finalization area A, it will infer the aggregation area as a circle, which is indicated by the large circle with the same center as the dot that indicates average location. If more data is available, the receiver may be able to instead use a rectangular shape to estimate the aggregation area.

introduces potential vulnurabilities, as discussed in Section 3; these vulnurabilities are part of the reason for introducing the finalization phase. An alternative approach to computing sums interactively would be to introduce granularity and count the different vehicles that agree with a particular status, an approach that is used in TBEV. To determine the best solution for this issue is very challenging because the effect of introducing granularity greatly depends on the amount of possible readings and the particular use case. Because SeDyA cannot in general use this solution because it is intended to be as generic as possible, but an application designer that works with a particular use-case can fix the granularity in the application, allowing him to use TBEV's approach.

FM and LC sketches will now be compared for their accuracy; indeed, [17] claims quite strong results for the LC sketches, but their analysis focusses on a limited use case (temperature in a relatively small sensor network). In SeDyA's setting, the amount of vehicles may range from a few dozen to several hundreds, each of which will add x different values to the sketch[3], so the amount of inputs can grow quickly. For this reason, a preliminary evaluation is performed to compare the accuracy of FM and LC sketches. There are two relevant variables that can be changed: the available bandwidth for data and the type of input, which is either binary (e.g., use the sketch for counting) or discrete in different sizes (e.g., use the sketch for sum). For this brief evaluation, only the amount of information contained in the aggregate is considered: no network simulation is used. The results have been generated using Java; for FM sketches, PCSA is always used. The estimates for the FM sketches have been generated using the optimized formula due to [29], as discussed in Section 3.1. To test the accuracy of a sketch, random input was generated and used to select bits in each sketch to simulate the random identities of vehicles for which the sketches will be used. The amount of identities and the values they reported were varied, to simulate low to high density scenarios; all the results are aggregated into a single figure for different available bandwidth, with the amount of hashed items on the x axis (participating vehicles times measurement size) and the relative error on the y axis. When PCSA is used for FM sketches, the size of each individual sketch as well as the amount of sketches used is relevant; different sizes are labeled as $size$x$number$. Each of the experiments was run five times with the same settings, but different random identities.

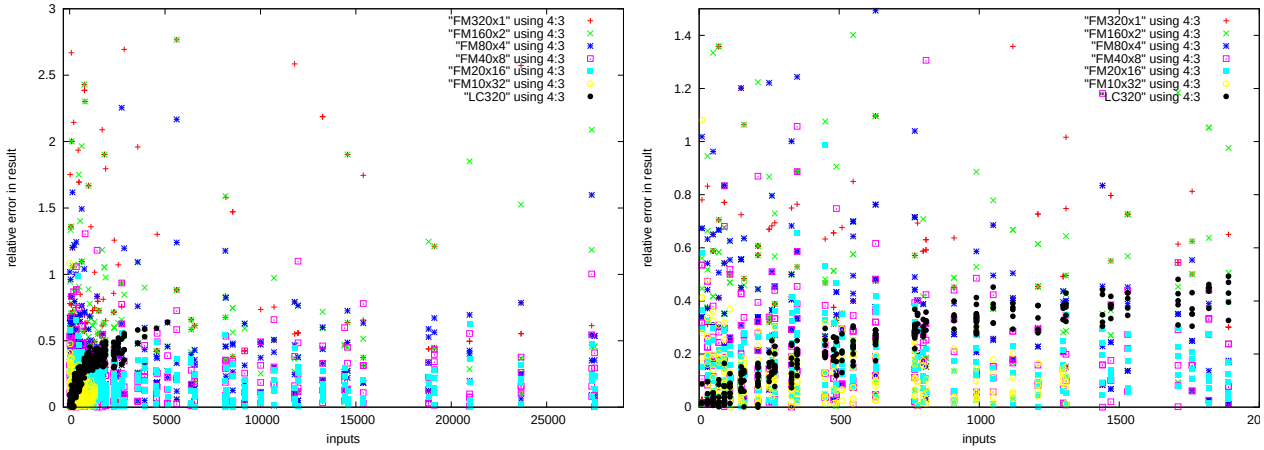
In [17], which claims a significant advantage for LC sketches, a sketch size of 320 bits is used. Figure 4.6a shows the accuracy for this sketch size. Note that LC sketches do not produce many results beyond 5000 inputs (e.g., 50 cars driving 100km/h). The cause of this is that an LC sketch will produce the result 'infinity' if it is full, as also noted in [17] as a shortcoming. Similarly, FM sketches that were too small to provide a reliable estimate, i.e., the maximum estimate of the sketch was less than the resulting value, were also excluded, which is why FM10x32 only appears on the bottom left of the graph. Figure 4.6b shows a more detailed image of the same graph, clearly indicating that for low input sizes LC sketches are more appropriate. However, as the amount of inputs increases, FM sketches are more useful. In addition, recall that FM sketches have the added advantage of having a geometric distribution, which allows them to grow on the fly. Out of all the FM sketches, it appears that the PCSA technique with more sketches results in more accuracy, as long as the sketch size can still count to a sufficiently high number.

Considering the fact that [17] does not consider security and therefore security overhead, a sketch size of 320 bits may be too large. Therefore, Figures 4.7a and 4.7b show similar measurements for a sketch size of 128 bits, and Figures 4.8a and 4.8b for 64 bits. For this size, LC sketches again show that they are useful for low inputs, due to their high accuracy, but for high inputs FM sketches remain much better. Thus, LC sketches will be used for any application that requires counting, while PCSA with FM sketches will be used for anything that requires a large amount of inputs, such as sum computations.

4.2.3 Overhead

The third issue in the aggregation phase is that of overhead, specifically in message overhead: when aggregates are more accurate, they are larger, requiring more bandwdith. Similarly, the additional security payload required (mainly signatures and certificates) is also significant. This is particularly important when many signatures must be produced, as is the case for the AM-FM approach. Overhead may also involve computational overhead; again, it is prudent to consider the amount of signatures a vehicle must generate

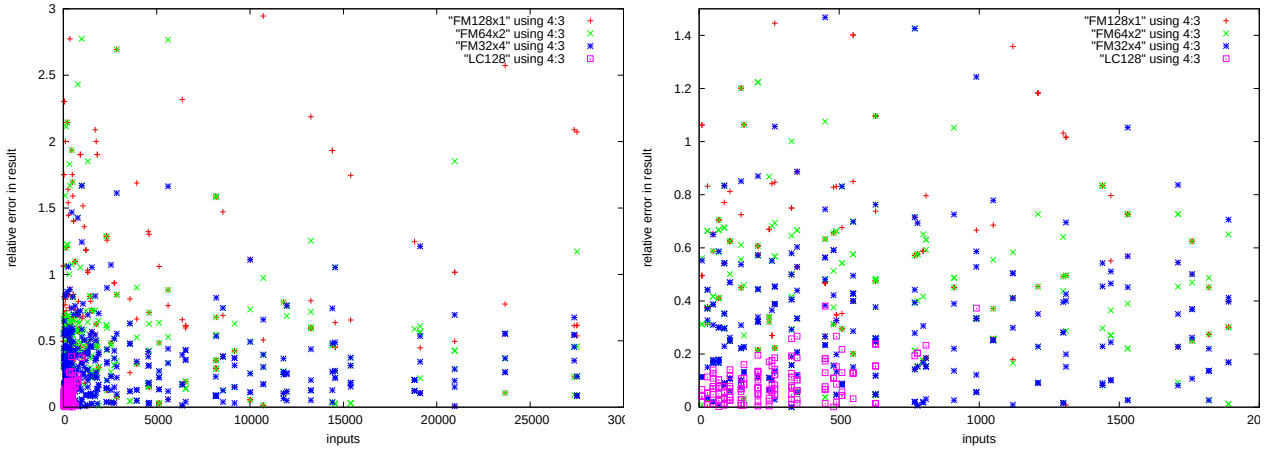
[3]Recall that sums are computed by hashing the values $(1, ID), (2, ID), ...(x, ID)$ to the sketch.



(a) Accuracy of sketches for a sketch size of 320 bits.

(b) A detailed view on the graph of Figure 4.6a.

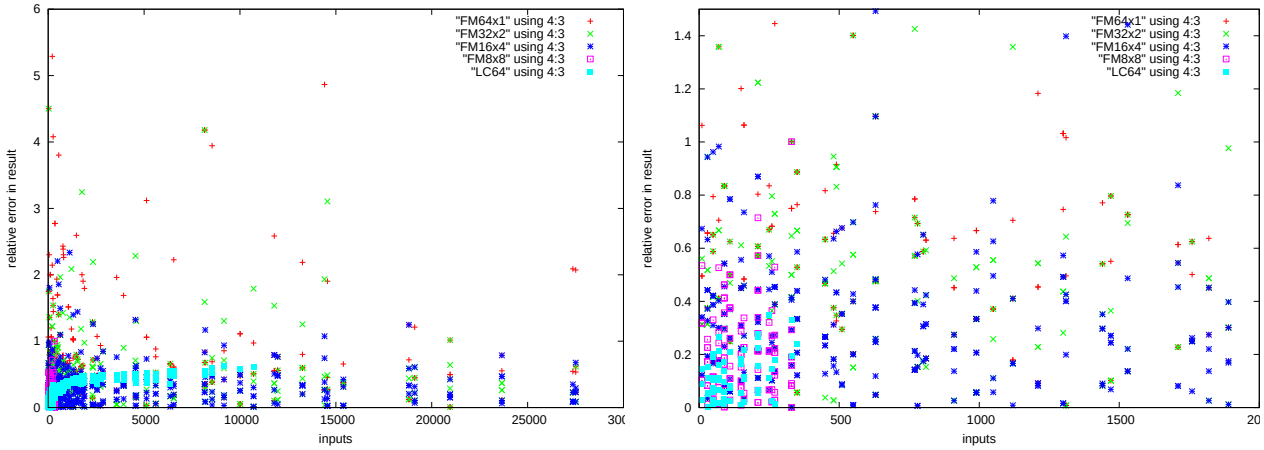
Figure 4.6: Relative error for varied amount of inputs and varied type of sketch when 320 bits are available for data (the value mentioned in [17]).



(a) Accuracy of sketches for a sketch size of 128 bits.

(b) A detailed view on the graph of Figure 4.7a.

Figure 4.7: Relative error for varied amount of inputs and varied type of sketch when 128 bits are available for data (the value mentioned in [17]).



(a) Accuracy of sketches for a sketch size of 64 bits.

(b) A detailed view on the graph of Figure 4.8a.

Figure 4.8: Relative error for varied amount of inputs and varied type of sketch when 64 bits are available for data (the value mentioned in [17]).

while evaluating the usefulness of such a scheme. The computational overhead for messages (signing and verification) can be quite high when software implementations are considered, even for regular VANET traffic, with verification speeds of several milliseconds on a high-speed desktop machine [35], which is both unrealistic hardware for an OBU and insufficiently fast for high traffic cases. However, cryptographic co-processors promise to make cryptography sufficiently fast for such applications. Because aggregation is not very delay sensitive, it should be able to manage with some delay as introduced by generating many signatures.

4.2.4 Discussion and conclusion

Instead of the segmented approach that is typically used in related work [17, 20], this thesis will focus on the case where flexible aggregation is used. This flexible aggregation is achieved by first choosing a low granularity distribution of points on a map, from which distances may be specified. Each time a vehicle transmits its observations as a sketch, it will use a map point to specify the origin from which the aggregation area is specified. The aggregation area itself is defined by additional sketches, that roughly estimate the distance from the map point. Depending on the situation (highway or urban scenario), the details are somewhat different, but in both cases the distance from map points is sufficient to define the location of the aggregation area. It is left for the beginning of the next phase to fix a border for the aggregation area. Note that while this introduces more inaccuracy into the aggregation mechanism, dynamic aggregation has the very important property that it will allow aggregation to scale despite expensive security mechanisms.

From the accuracy results from the previous section, it is clear that both AM-FM sketches and AM-LC sketches can be used in a VANET aggregation scenario, for summation and counting, respectively. This will allow the most effective usage of the available bandwidth.

Finally, it is worth repeating that in [20], it is considered to be somewhat too inaccurate to compute sums in the way proposed here. The work instead replaces the summation with an iterative mechanism to count nodes that agree with a certain range. However, this is a challenging approach to accomplish in VANETs, because nodes are much more dynamic in their behavior, and an estimate of the amount of participating nodes is unknown to the receivers of the aggregate. It may be possible to transfer these ideas by introducing granularity to convert the sketches that compute sums to sketches that only count. However, this will introduce more inaccuracy, because nodes can no longer freely add values but can only state agreement or disagreement with a particular statement, such as 'The current average speed is 50km/h!'. In terms of security, this is beneficial, because the attacker can now only flip one bit, limiting the impact of an attack. An alternative to SeDyA could attempt to exploit this approach; however, as noted previously, the goal of

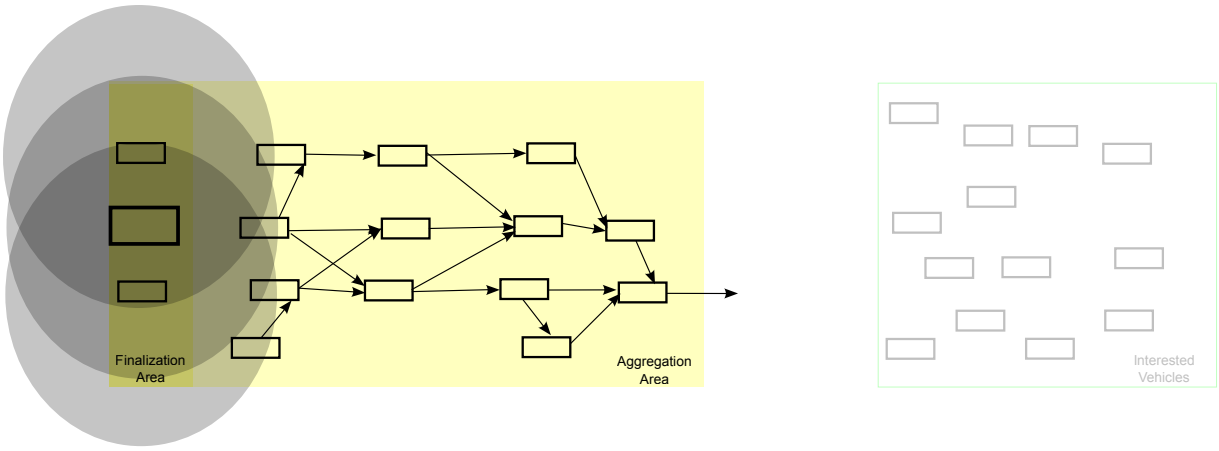


Figure 4.9: The finalization phase in the context of the aggregation model.

SeDyA is also to be flexible.

4.2.5 SeDyA's solutions

To summarize, the aggregation phase solves the problem of aggregation with both FM and LC sketches, using them for summation and counting, respectively. To have a dynamic aggregation area that can still be secured, additional sketches are employed to describe the distance from a fixed point that is common for all vehicles in the aggregation area. The benefit of aggregating in time segments is considered negligible, so fixed time segments will be used to save space. For security purposes, signatures that can be aggregated are employed. Because it is still necessary to merge different sketches, signature aggregation is only performed on those signatures associated with the initial sequence of 1-bits; all other signatures are included separately. Each signature signs a position in the sketch, the fixed time segment, the fixed point and any additional information that is constant throughout the aggregation area (that is, things like protocol version). The transition to the next phase is left for the next section; however, to define the aggregation area, this transition must include the location of the vehicle that performs it.

4.3 Phase 2: Finalization Phase

In the second phase of SeDyA, the main goal is to determine when an aggregate is complete, and how the implicit finalization of this aggregate can be made explicit for security purposes. In the previous phase, the aggregation phase, SeDyA aggregates in a relatively small, local setting to find locally stable information such as average speed in a traffic jam. In the next phase, the dissemination phase, SeDyA will disseminate the finalized aggregate and provide the data to the application, which can then use it to infer information such as the fact that there is a traffic jam. The current phase determines the security of the resulting aggregate and thus determines the quality of the information that the interested vehicles receive.

This section will discuss the process of finalization, which determines the starting point of this phase, followed by a discussion of how to generate the multisignature that is used as the main security mechanism in SeDyA. The main issue in this phase is how the latter; how can a multisignature be generated in a distributed fashion, without reliable communication links? An additional issue that closely relates to this is the time required for a message to pass through the finalization phase, which should not require waiting for every vehicle to perform a task.

4.3.1 Finalization

The first issue considered here is when finalization of an aggregate should occur. In current work, finalization is implicitly achieved, because of the fixed aggregation area and thus bounded amount of messages in the aggregation area (e.g., 500 meter stretches of road in SAS). However, with the introduction of dynamic aggregation areas in SeDyA's aggregation phase, implicit finalization will no longer work, because an aggregate may now grow much more. Thus, explicit finalization serves to fix the aggregation area. In addition, explicit finalization provides the basis for the security mechanism that will be introduced later in this section. However, the issue of when to finalize a message remains. There are many approaches to determining when a particular value is locally stable or should otherwise be considered a final message. However, note that a simple threshold may not be sufficient, because some types of information graduately increase or decrease over area, or the data does not segment the road into regions where the value is high or low[4]. Finally, the initial data will not concern the whole aggregation area, so finalization should not occur until most of the information is disseminated throughout the area. However, designing and implementing these types of parameters is complex and highly application-specific. Therfore, for the remainder of this thesis, a combination of a minimum time spent in the aggregation phase and a simple threshold will be used. The threshold, henceforth referred to as edge threshold, indicates whether the vehicle is on an edge between two aggregation areas based on the beacons of surrounding vehicles. The vehicles are divided into two groups (ahead of and behind the vehicle); the edge threshold determines what the difference between the averaged values reported by these two groups must be, for the vehicle to be considered an edge node.

Finalized message selection is the first step of this phase, where one or more vehicles will detect a message that should be finalized. These vehicles, called edge nodes, will be either close in range to each other (e.g., both on the same end of a traffic jam in a highway scenario) or widely distributed (each on a different end of a traffic jam, or an urban scenario). Each edge node will broadcast the aggregate, its signature on the aggregate, its certificate, its location and a signature on its location. Note that all vehicles receiving this message should check that the location lies within their communication range; if this is not the case, then they should report an attempted attack and ignore further finalization messages from its sender (identified by network address). Therefore, if the attacker attempts a replay attack in the same time slot, or attempts to finalize with a faked position, she may be detected and blacklisted. The finalizing vehicle is then selected by choosing the message with the lowest identfier, and its message is then the finalized message. Each edge node re-broadcasts this finalized message, after which the signature collection protocol will begin. If the attacker colludes with other vehicles[5], she may be able to convince the network to use her message for the signature collection protocol. However, because each vehicle will verify the contents of the message, the attacker can either not have a large impact or only obtain very few signatures, which means that the attacker is no better off than to send a message on her own behalf.

The message format is the same for both the finalization and the signature collection protocol: $\langle Agg, loc, P_1, \sigma_m, S_m, \sigma_v, C_v \rangle$. These represent the aggregate data, the location of the finalizing vehicle, the security payload from the aggregation phase, the multisignature, the set of certificates, the vehicles' signature and the vehicles' certificate. The aggregate data includes the sketches for the aggregation area, the data itself, the time slot and any other application-specific constants. All signatures in phase 2 will be generated on (Agg, loc), thus excluding P_1, which will be discarded at the end of this phase.

The main purpose of this phase, is to provide additional security guarantees. As discussed in Chapter 3, the use of sketches provides the capability to guarantee integrity of different observations even when they are aggregated into an aggegate message. However, it has also been shown that this leads to some potential attacks from an adaptive attacker that performs an aggregation step, in particular when computing sums, due to the fact that an attacker can sign more than one bit in these sketches. Recall that the attacker has full control over her own observation, which can be generated based on the aggregates she merges. The attacker can arbitrarily inflate or deflate the aggregate, but such an attack can be detected by neighboring vehicles, due to the fact that the estimate of the aggregate is no longer consistent with the real world. Therefore,

[4]For example, consider a traffic jam; there is no hard distinction between vehicles being 'in' or 'out of' the traffic jam; there is a lot of slow-moving traffic before and after it.

[5]For example, by using an exploit in the software of the surrounding vehicles.

vehicles should always check a received aggregate against their own knowledge; however, recall that sketches come with significant inaccuracy, on top of the fact that the aggregate already provides an average of several values, so this is not a perfect solution. In particular, recall that an attacker may modify her observation to have exactly the desired impact. Potentially, an attacker can indeed have a very large impact that cannot be detected by, for example, an over-representation of the attackers' identity after the aggregate has been generated. In addition to potential attacks like this one, an attacker may initialize the finalization process at an early stage, or an aggregate may simply contain a sketch that does not appropriately represent reality due to the probabilistic nature of sketches. These issues are all defeated by the finalization phase by giving each vehicle in the aggregation area the opportunity to contribute a signature to the finalized message. When a vehicle attaches its signature to such a message, this signature is used as evidence that the vehicle agrees with the message. Agreeing with a message can be decided in many ways, but SeDyA will use a simple threshold, henceforth called the agreement threshold, that checks its own value and the average of its current neighbor table against the aggregate to determine whether to sign. Because attaching signatures for every vehicle consumes a linear amount of space, and the size of a signature is significant, a multisignature scheme will be used to compress all signatures into a single signature. The space consumed will still be linear, because the certificates of all participants are required to verify the message. As noted by [21], it is not information-theoretically possible to avoid including some indication of who signed what. The issue of certificates and their size is addressed in the next chapter. However, before size becomes an issue, there must first be an effective method to collect the signatures and certificates.

4.3.2 Signature collection protocol

This is the problem that will be solved by the signature collection protocol that is presented below. Signatures are collected using a protocol similar to a broadcast dissemination protocol, such as the distance-based scheme in [47], where the distance from the broadcasting vehicle and the transmission backoff of the current vehicle are directly related. In a broadcast protocol, the delay grows as the vehicles are closer together; in the signature collection protocol, the exact inverse happens. This means that the closer two vehicles are, the faster the receiving vehicle re-broadcasts the message with the addition of its own signature. Therefore, vehicles further away will obtain the signatures of most vehicles inbetween the previous sender and themselves. This is illustrated in Figure 4.10. Vehicles can use beacons to estimate the amount of vehicles between the original sender and themselves; a vehicle should schedule its transmission to occur after that of the intermediate vehicles to avoid collisions. To prevent flooding and make scheduling easier, only a small fixed amount of messages should be sent per beaconing interval.

Recall that a message is formatted as $\langle Agg, loc, P_1, \sigma_m, S_m, \sigma_v, C_v \rangle$, representing the aggregate data, the location of the finalizing vehicle, the security payload from the aggregation phase, the multisignature, the set of certificates, the vehicles' signature and the vehicles' certificate, respectively. After a vehicle A receives such messages in sequence from three vehicles B, C, D, represented in Figure 4.10 by a dotted, dashed and solid line respectively, it will have computed $\sigma_m \cdot \sigma_B \cdot \sigma_C \cdot \sigma_D$. More precisely, after receiving the message from B, A will compute $\sigma_m \cdot \sigma_B$; after it receives the message from C, it will compute $\sigma_m \cdot \sigma_B \cdot \sigma_C$ and so on. This applies only when the certificate sets S_m (and thus the multisignatures σ_m) are the same for each received message. If the certificate sets differ, but only by certificates of vehicles from which A received a message, then A will still have the individual signatures. For example, consider the case where B transmits first, with σ_m and σ_B, then C sends with σ_m and σ_C. Now, D sends $\sigma_m \cdot \sigma_B$ and σ_D. If A receives such messages, it can still compute $\sigma_m \cdot \sigma_B \cdot \sigma_C$ from the available signatures. However, when the certificate sets differ by a vehicle from which A has not received a message, then A may compute its σ_m by multiplying the conflicting multisignatures in addition to the signatures of the neighbors (i.e., $\sigma_{m1} \cdot \sigma_{m2} \cdot \sigma_B \cdot \sigma_C$). The disadvantage of this solution is that the certificate set will be a multiset that grows faster as this type of problem occurs more often. However, the inclusion of the individual signature (rather than only sending $\sigma_m \cdot \sigma_{ID}$) and the scheduling should limit the necessity of this solution to a minimum.

This protocol allows most runs to successfully provide a number of signatures that is close to the amount of vehicles in the area, for an aggregate that is legitimate. There is always the risk that loss will cause some missing signatures, as well as the risk that vehicles have left the aggregation area since their contribution to

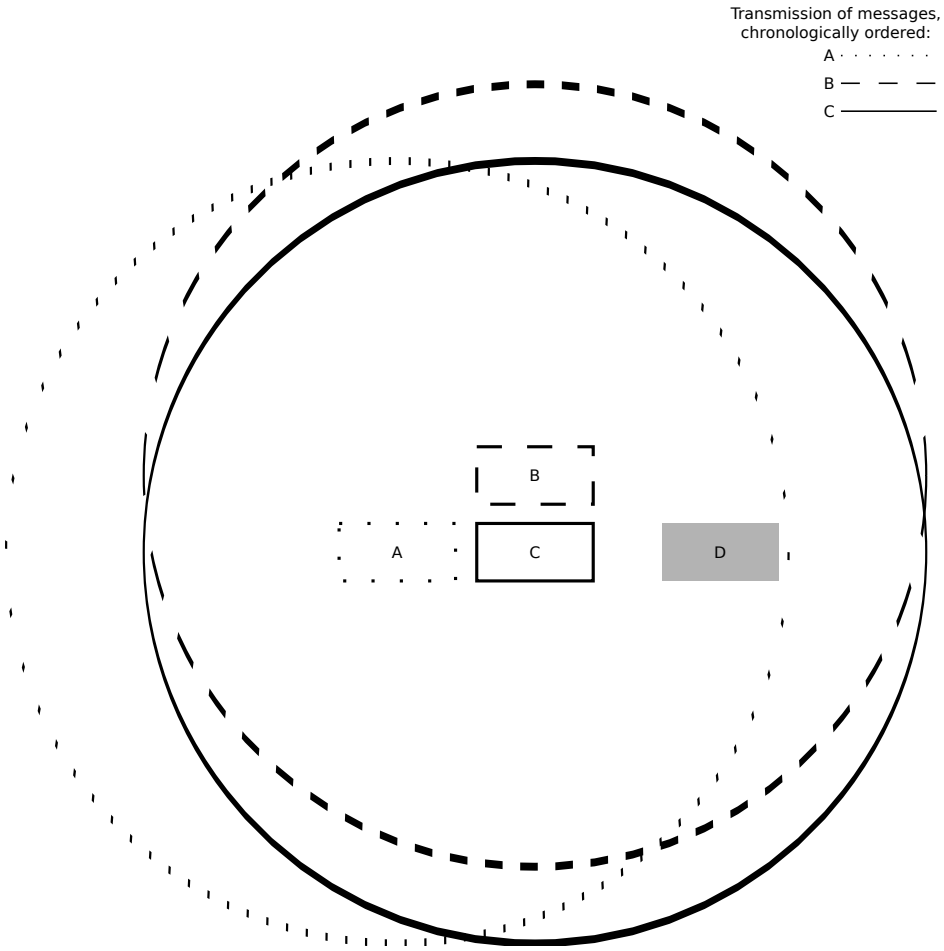


Figure 4.10: This figure shows four vehicles, where three have broadcasted the message with their signature. The broadcasts are indicated by circles; the dotted line is the oldest broadcast and the line is the most recent one. The right-most vehicle receives the dotted broadcast first, at which point it queues a message with $\sigma_m \cdot \sigma_{dotted}$. It then receives the dashed broadcast and updates to $\sigma_m \cdot \sigma_{dotted} \cdot \sigma_{dashed}$. Finally the line broadcast is received, which contains the multisignature $\sigma_m \cdot \sigma_{dotted}$. The right-most vehicle knows σ_{dotted}, so she can compute $\sigma_m \cdot \sigma_{dotted} \cdot \sigma_{dashed} \cdot \sigma_{line}$.

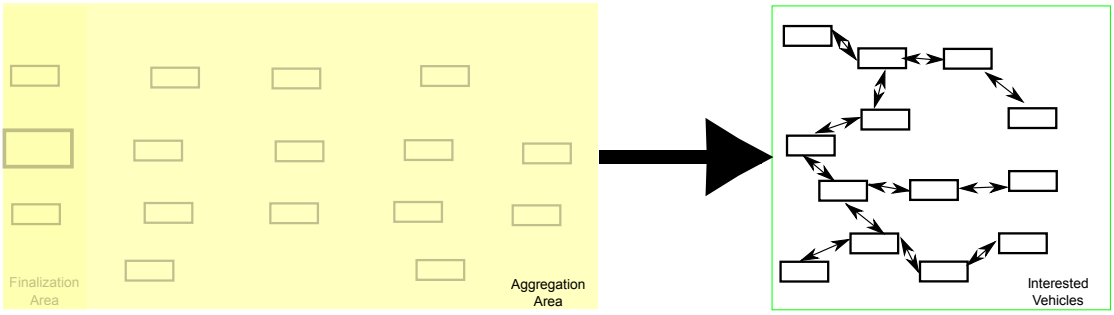


Figure 4.11: The dissemination phase in the context of the aggregation model.

the aggregate. Finally, it may be considered desirable to further limit the amount of certificates. This can be achieved by allowing the finalizing vehicle to select a signing policy that specifies with which probability a vehicle should include its signature.

4.3.3 SeDyA's solutions

In summary, the finalization phase performs the most important tasks with respect to security. The transition from the aggregation phase works by detecting that the message is complete when compared with the beacons of the neighbors, followed by a broadcast of the finalized message by this vehicle and any nearby vehicles that have come to the same conclusion. Security is increased in this phase by forwarding the finalized message through the aggregation area and allowing vehicles to add their signature when they agree with the message. To save resources, but retain the advantages of these explicit signatures, multisignatures are employed to compress signatures into a single one. To collect the signatures for generating a multisignature, the signature collection protocol is used, which operates in parallel with forwarding the message through the aggregation area. Therefore, as the message is forwarded, the multisignature is generated at the same time. The transition to the next phase is not explicit. When a vehicle detects it is outside the aggregation area, it can drop the security payload from the aggregation phase and continue with the dissemination phase.

4.4 Phase 3: Dissemination Phase

In the third and last phase, the dissemination phase, the goal is to distribute the finalized messages and associated multisignatures and certificate sets to interested vehicles beyond the bounds of the aggregation area. This is illustrated in Figure 4.11. The interested vehicles can then use this information for a number of aggregation applications such as traffic statistics, traffic routing, finding free parking spaces and traffic jam detection. For such applications to operate properly, the information must be accurate and protected from tampering by attackers; different approaches for this detection is the main challenge in this phase. The dissemination process in this phase is trivial; any efficient broadcast dissemination scheme will perform reasonably.

Recall from the aggregation phase that there is now a dynamic aggregation area, which varies in size depending on the particular aggregate. This makes defining remote impersonation attacks slightly more difficult, because such attacks are, strictly speaking, any vehicle that contributes to an aggregate from outside the aggregation area. Now that the aggregation area no longer has a fixed bound, a remote impersonation attack is no longer defined before aggregation is finalized, because the bound is probabilistic. However, in general SeDyA is particularly interested in vehicles that are relatively far from the aggregation area; protec-

tion against attacks by vehicles just outside the aggregation area can be prevented through the measures in the other two phases.

Note that it is reasonable to assume that the attacker must have a significant impact on the aggregate in order to consider her attack successful, because the probabilistic nature of aggregates make small modifications irrelevant. The attacker will require the message to not only carry the modified aggregate, but also that the receivers will have confidence that the message is legitimate. Without these properties, an attack does not help, and these two are exactly the properties that SeDyA aims to take away in its first two phases. Thus, part of the security of SeDyA involves how interested vehicles will treat the information they are sent.

In addition, it may be desirable to be able to merge two finalized messages into a larger one (e.g., consider a traffic jam with a small gap where vehicles drive twice the speed). This is easy when finalization is implicit, but an attacker can abuse this legitimate use case to modify the aggregate even when she is not in the aggregation area. Each of these issues is discussed seperately below, along with an indication of how to solve them.

4.4.1 Confidence

The main issue in this phase is the confidence an interested vehicle can assign to the contents of message. SeDyA's first two phases can only provide so much; in the end, it remains the case that the attacker is in possession of key material, which may be used to transmit malicious information with valid signatures. This cannot be changed by any cryptographic mechanism; the best that can be done is to guarantee maximum detection of malicious data and to limit the key material available by revocation and legal protection. The question remains, how can the confidence in any message be quantified, such as to be useful to the interested vehicle that is attempting to estimate the value of a message she just received. As already noted in the discussion of SeDyA's second phase, the amount of certificates attached to the received message should play an important role here. In addition, note that the message will contain a sketch that estimates the amount of participating vehicles for the purpose of computing an average; while somewhat inaccurate, this can be used to compare against the amount of signing vehicles. Freshness of the message is another useful parameter that can be used to indicate how confident the reciever of the message should be about its content. Finally, the detection of attacks will play a large role in the quantification of confidence, as the likelyhood of an attack is directly associated with how trustworthy the message is.

Confidence quantification can also be used to provide the necessary bounds on the dissemination phase, by avoiding the selection of messages with low confidence. Given infinite bandwidth, it does not make sense to limite the available bandwidth for each message. Since the available bandwidth can vary greatly over the range of interested vehicles, the most convenient approach is to assume infinite bandwidth and make a best effort to disseminate the most useful information. This usefulness can be determined directly by a vehicle's confidence in a message. However, it is out of scope for SeDyA to introduce a new best-effort broadcast dissemination scheme, the parameters above should provide some insight in the usefulness of such a scheme. SeDyA assumes that an arbitrary protocol is used to disseminate the messages as far as possible given bandwidth constraints. The only requirement made for such a scheme is that each vehicle should at least verify that the sender of the message and the aggregation area are roughly on the same side of the vehicle. The lack thereof indicates that a remote impersonation attack may be in progress.

4.4.2 Attack detection

The detection of such remote impersonation attacks is one of the basic heuristics that SeDyA uses to detect attacks. In general, it is difficult to evaluate whether an attack is in progress based on messages alone, as shown by the continued research effort in intrusion detection systems. However, in VANETs there is additional information available; physical models provide the initial sanity checks to detect absurd statements such as speeds of thousands of kilometers per hour. This approach of incorporating different sources of information and using them to detect attacks is fundemental to the field of data consistency, which has recently gained interest from the research community. However, instead of developing an entirely new method of performing data consistency, SeDyA may be used as an information source in such a scheme.

SeDyA offers some useful additional metrics to detect attacks, which are non-trivial to obtain in a generic data consistency setting. Note that data consistency checks should occur in every phase, although SeDyA can only be used as input in the dissemination phase.

Vehicles that are disseminating a finalized aggregate should check the contents of such an aggregate against the real world, detecting any attacks that contradict with the physical model. This includes the source of the message, which is especially useful in highway scenarios; a message that is being forwarded by vehicles behind, but contains claims about an area in front, is suspicious. Another obvious source of an attack is two conflicting views about the same region, or different messages for that same region. Because a vehicle should still forward information, and it cannot always determine which of the two is the malicious message, it can forward both messages together in this situation.

4.4.3 Further aggregation and its challenges

Finally, despite dynamic aggregation, vehicles may want to merge the different aggregated messages they receive into a single one. For example, it may happen that two traffic jams are close together, but there is a gap of about one kilometer. While this operation is trivial in an aggregation scheme that does not consider security, a secure solution is more challenging, because the very purpose of such a solution is to keep two distinct finalized messages separate. Existing work on secure aggregation does not provide a solution for this issue. For example, if SAS used, then merging two implicitly finalized messages about different road segments would merge two sets of proofs with different inputs to their cryptographic signatures. As a consequence, the receiver is no longer able to verify the messages, because she cannot determine which content is signed with which fixed piece of information.

The same issue exists with SeDyA, but it will occur much less frequently due to the dynamic aggregation mechanism. In this setting, the only further 'aggregation' possible after the finalization of the message is appending more certificates and signatures to the message. In theory, it is possible that an attacker has network access both inside the aggregation area and on the path between the aggregation area and the interested vehicles. This attacker can exploit the possibility of adding signatures by attacking the aggregation phase by modifying the aggregate only by a limited amount, such that some vehicles in the aggregation area will still agree to sign it. She can then use her network access on the path to add additional signatures with newer pseudonyms, as they become available, to increase the confidence in this message. However, such attacks are not very relevant, because they rely on a number of unrealistic assumptions and the impact of the attack is insignificant, unless the expected amount of signers for legitimate messages is very low. The most notable assumption necessary for this attack is that the attacker needs sufficient knowledge about the aggregation area to predict the behavior of participating vehicles. It is possible for the attacker to generate an attack with higher impact as she obtains possession of more key material; however, the best that such an attacker can do is induce conflicting world views in the interested vehicles, because of the honest majority assumption.

One possible approach to achieve aggregation beyond the bounds of the fixed points regardless of these issues is to allow vehicles to consider aggregates they receive as an observation. However, this severely hurts the mechanisms that detect attacks, because vehicles that are not in an aggregation area may now suddenly make claims about that area. The only feasible approach is to allow application designers to create specific statements, such as 'in the next twenty kilometers, there is no traffic jam', and perform binary counting on them. This approach is similar to that of TBEV, but it does not take advantage of the possibility of dynamic aggregation that SeDyA offers and still risks attacks.

4.4.4 SeDyA's solutions

This final phase of SeDyA can use any existing dissemination protocol to forward messages throughout the network. The main purpose of aggregation is to allow the rest of the network to learn the state of each aggregation area. Each vehicle that receives a finalized message can compute its confidence in the message and determine how useful it is: for this purpose SeDyA specifies the amount of signers, the proximity of the aggregation area and the freshness of the message. For the dissemination of the messages, SeDyA does

not put constraints on forwarding but instead optimizes by re-arranging the forwarded messages, such that the most useful messages are forwarded first. This is determined by the confidence that was assigned to the message; another measure that will be used is the attack probability, which can be determined by comparing the amount of signers with the considered amount of vehicles in the area and the size of the area.

4.5 Summary

This chapter has discussed the ideas and goals that SeDyA aims to achieve, including a high-level overview of how SeDyA works and its relation to related work. SeDyA splits the process of secure VANET aggregation into three parts, which it solves with individual solutions that each contribute to an increased level of security. The aggregation phase shows how to obtain agreement within the aggregation area about the data and the bounds, while the finalization area shows how to finalize this decision and generate a multisignature that shows that the process was indeed performed correctly. The dissemination phase discusses how to forward the messages through the rest of the network, what kind of guarantees are offered and how those guarantees relate to the previous phases. Finally, the dissemination phase also explains how to apply SeDyA by discussing factors that contribute to attack detection and that provide confidence metrics that may be used in data consistency mechanisms. What remains now is to determine how to implement SeDyA efficiently, using cryptographic schemes to provide the primitives that were assumed in this chapter.

Chapter 5

SeDyA: implementation

This chapter discusses implementation issues in SeDyA, consisting of cryptographic schemes and a more detailed discussion of bandwidth consumption and certification authorities. It will begin by introducing the necessary cryptographic background, discussing every option that is considered for securing SeDyA. This will be followed by a selection of cryptographic schemes for the aggregation and finalization phases from the previous chapter and a discussion of the alternatives. Subsequently, the first two phases of SeDyA will be reviewed in light of the implementation issues and the chapter will conclude with a complete overview of SeDyA that includes the implementation issues. The dissemination phase is not discussed; secure and efficient dissemination of the resulting messages that SeDyA produces is an entirely different field. For the implementation used for the evaluation, simple flooding with a forwarding probability of $\frac{1}{2}$ is used, analogous to the first phase.

5.1 Cryptographic Background

This section will introduce the necessary cryptographic background to understand the details of SeDyA. This includes the introduction of elliptic curve and bilinear pairing-based cryptography, each with a discussion of the different cryptographic schemes that may be used in SeDyA: ECDSA and ECQV ECDSA for elliptic curves; IBMS and IBAS for pairing-based cryptography.

Recall SeDyA involves three types of cryptographic signatures; regular signatures, multisignatures and aggregate signatures. The definitions are repeated here:

Signature For a regular one signer generates a signature on one message using her private key. The signature can then be verified by anyone that has posession of a legitmate copy of the corresponding public key.

Multisignature Given a group of n participants, each with their own keypair (pk_i, sk_i) a multisignature on a message M can be generated by any subset of all participants L by computing it from the individual signature that each participant generates on M. Then, the signature can be verified by using all the public keys of the participants in L and computing from them the 'public key' of L.

Aggregate signature Given a group of n distinct participants and n distinct messages, an aggregate signature is a signature that can be computed from the signatures that are generated by each participant on its own message.

5.1.1 Elliptic Curve Cryptography

This section will discuss some of the basics for elliptic curve cryptography (ECC), giving an intuition of how a curve is used and which hard problems play a role. This discussion forms the basis for the subsequent sections that discuss identity-based encryption (IBE) and (multi)signatures, both of which require knowledge of ECC basics.

An elliptic curve is essentially a function with two parameters, which can be written as[1]: $y^2 = x^3 + a \cdot x + b$. Here, a and b are the values that define the shape of the curve, and thus which points will be on the curve. Each point is simply (x, y); the operation on a curve is point addition (as opposed to multiplication in RSA). The way to obtain these points differs for each curve definition; an overview can be found on the webpage provided in the previous footnote. For cryptography, one typically uses carefully chosen elliptic curves over finite fields, i.e., $x, y \in \mathbb{F}_q$, typically written as $E(\mathbb{F}_q)$. It can be proven that the points on an elliptic curve over finite fields always forms a cyclic group [31], which can be used for cryptographic purposes by defining a discrete logarithm problem over it. Given that $P \in E(\mathbb{F}_q)$ is a point that generates the curve, the discrete logarithm problem (DLP) can be written as finding $a \in \mathbb{Z}$ given aP, P (analogous to the problem in fields; finding $x \in \mathbb{Z}$ given $g^x, g \in \mathbb{Z}_{ıı}$). Similarly, the computational and decisional diffie-hellman problems (CDH and DDH) can be written as finding abP given P, aP, bP and determining whether $c = ab$ given P, aP, bP, cP respectively. These are the hard problems on which the diffie-hellman key exchange is built, among many other public key cryptography schemes, and they often form the basis of a security proof.

ECDSA

One of these sechemes is the Digital Signature Algorithm (DSA). Elliptic curve cryptography can be applied to provide signatures using the Elliptic Curve Digital Signature Algorithm (ECDSA). ECDSA is the standard signing algorithm for VANETs, as stated in IEEE 1609.2 [1]. The algorithm is briefly repeated here to illustrate its operation[2].
Setup Select a generator P of group G, set $n = |G|$ and a hash function $H : \{0,1\}^* \rightarrow [1, n-1]$.
Key Generation is performed by taking a random private key $d \in [1, n-1]$ and a public key $PK = d \cdot P$.
Signing of a message M is performed by computing $z = H(M)$, selecting a random value $k \in [1, n-1]$ and computing $r = (k \cdot P)_x \mod n$, where the $_x$ signifies taking the x component of the point on the curve. Finally, compute $s = k^{-1}(z + r \cdot d) \mod n$ and let the signature be (r, s).
Verification of signature $\sigma = (r, s)$ can be performed by again computing $z = H(M)$. Now, take $w = s^{-1} \mod n$, compute $u_1 = z \cdot w$ and $u_2 = r \cdot w$ (both $\mod n$). Finally, compute the point $R = u_1 \cdot G + u_2 \cdot PK$ and check that the x-coordinate matches: $R_x = r$.

ECQV ECDSA

The Elliptic Curve Qu-Vanstone (ECQV) implicit certificate scheme is one of several implicit certifcation schemes that is commonly combined with ECDSA signatures. In principle, implicit certificates replace the actual certificate by an identity and a value, as opposed to an explicit certificate which has an idenity, a signature and a public key. The implicit certificate can be used directly for encryption (or signing) and its most important advantage is the size of the (implicit) certificate is smaller, saving bandwidth in scenarios where many public keys are not known to the receiver. The setting is the same as ECDSA, but the key generation phase is now as follows, interactively between the CA and the user. The CA key pair is generated as any other key, with random c and $C = cP$, same as the regular ECDSA setting. Note that in ECDSA, the CA was not discussed, because certification happens by regular X.509 means; however, ECQV changes this. The user generates a certification request $R = rq \cdot P$, using a random value rq. The implicit certificate can be computed as $IC = R + k \cdot P$, with an implicit signature $sig = c + H(IC, ID) \cdot k$ and a random k such that $k \in [0, n]$. The key pair is then determined as $sig + H(IC, ID) \cdot rq$ as private key and $C + H(IC, ID) \cdot IC$ as public key. Here ID is the identity of the user; the choice for k is made by the CA. The key pair is computed by the user, while the implicit certificate and signature are computed by the CA. Now, the public key is implicitly certified, meaning that anyone who generates a signature that verifies with this key is in possession of the corresponding private key. Note that when transmitting a message, the user should send

[1] This is the Weierstrass curve; others, such as Montgomery or Twisted Edwards curves exist. Some curves may be rewritten to others and some are computationally cheaper than others; see `http://hyperelliptic.org/EFD/g1p/index.html` for a comprehensive overview.

[2] An extensive description may be found in the standard ANSI X9.62 and FIPS 186-3, which discusses appropriate curves and similar details.

the implicit certificate and the signature she generates, **not** the public key and the signature, because anyone can re-compute the public key from the certificate and the identity of the user.

When verifying many signatures in a short time, both ECQV ECDSA and ECDSA can still be too much for a regular OBU's processor. However, cryptographic co-processors are capable of doing the required operations (one inversion mod n, 3 scalar multiplications +1 for reconstruction, 1 point addition +1 for reconstruction) in less about 1 ms, assuming that scalar multiplication is the most expensive operation [30].

5.1.2 Pairing-based Cryptography & Identity Based Encryption

Just like RSA primes, elliptic curves must be carefully chosen to build a secure system. One of the issues encountered when using elliptic curves is that of pairings[3]. These pairings map two points of an elliptic curve to a group G using a bilinear map. The map is of particular interest because another class of cryptosystems can be built on them; such a map is defined as $e : E(\mathbb{F}_q) \times E(\mathbb{F}_q) \to G$ and respects two important properties: $e(P + Q, R) = e(P, R)e(Q, R)$ as well as $e(P, Q + R) = e(P, Q)e(P, R)$ for all points $P, Q, R \in E(\mathbb{F}_q)$ (bilinearity) and for all $P \neq 0 : \exists Q \in E(\mathbb{F}_q) : e(P, Q) \neq 1$ (non-degeneracy). Note that bilinearity implies that $e(aP, Q) = e(P, Q)^a = e(P, aQ)$, a useful property that is exploited by a variety of cryptosystems. One of the classes of elliptic curves on which pairings with these properties can be defined are called supersingular curves.

Note the properties of pairings obviously defeat cryptosystems based on these curves that require hardness of the DDH problem on elliptic curves; it is now simple to check that $e(P, cP) = e(P, abP)$ by testing against $e(aP, bP)$. However, a new problem can now be defined, the bilinear diffie-hellman problem (BDH): find $e(P, P)^{abc}$ given P, aP, bP, cP. Using this problem, we can implement identity-based encryption (IBE), where the idea is to replace the public key of a user with a name or address. After Boneh et al. [7] introduced a practical IBE encryption scheme on pairings, many have followed to implement improvements and adjustments, including signatures based on IBE, certificateless encryption and certificate-based encryption. The general structure of all these schemes follows that of the original scheme. This scheme contains a single trusted third party (the CA), which issues private keys based on the name of the user, or some other unique identifier, such as a hash of name and validity period. The public key is determined using a hash function and the public key of the CA; decryption is performed using the issued private key and the public key of the server.

Basic IBE consists of four phases; setup, private key extraction, encryption and decryption.

Setup The CA selects the parameters, including a generator P for the group of the curve $\langle P \rangle$, the group of the map G, two hash functions, a master key s and computes its public key sP. The hash function $H_1 : \{0,1\}^* \to \langle P \rangle$ and $H_2 : G \to M \in \{0,1\}^n$.

Extract key The CA computes a key d_I from identity $I \in \{0,1\}^*$ as $d_I = sH_1(I)$, sent to the user over a secure channel.

Encrypt Encryption of $M \in \{0,1\}^n$is performed by computing $mask = H_2(e(H_1(I), sP)^r)$ the ciphertext $C = (rP, M \oplus mask)$.

Decrypt Decryption of[4] (U, V) also performed by computing $mask$: $H_2(e(d_I, U))$ and then $M = V \oplus mask$.

This works because $e(d_I, U) = e(sH_1(I), rP) = e(H_1(I), P)^{rs} = e(H_1(I), sP)^r$, which is exactly what was used as input of an encryption operation. Notice that breaking this is indeed the BDH problem: P and sP are known, rP is transmitted as part of the message and $H_1(I) = aP$ (for some a, because the group is cyclic) is the public key of the user. To decrypt the message, one must compute $e(P, P)^{sra} = e(H_1(I), sP)^r$ from these four values; this is exactly the BDH problem.

Full IBE adds protection against adaptive chosen ciphertext attacks. This scheme works for similar reason as the basic scheme; it just adds some additional randomness. The additional protection is implemented as follows (key extraction remains the same):

Setup Select two more hash functions, $H_3 : \{0,1\}^n \to \mathbb{Z}_q^*$ and $H_4 : \{0,1\}^n \to \{0,1\}^n$.

Encryption Choose $\sigma \in \{0,1\}^n$ and compute $r = H_3(\sigma, M)$. Then compute the ciphertext as $C = (rP, \sigma \oplus mask, M \oplus H_4(\sigma))$, where $mask$ is computed as before.

[3]Various types exist; most common are Weil and Tate pairings.

[4]Note on notation: (U, V) is used to clearly indicate the knowledge used while decrypting.

Decryption Using ciphertext (U, V, W), compute $\sigma = V \oplus H_2(e(d_I, U))$ and $M = W \oplus H_4(\sigma)$. Also, verify that $U = H_3(\sigma, M)P$.

Identity based signatures To generate signatures from an IBE scheme, extend the basic IBE scheme of Boneh et al. with a third hash function $H_3 : \{0,1\}^* \times \langle P \rangle \to \mathbb{Z}_q^*$. A signature on message M is generated as follows: set $U = rH_1(I)$, $h = H_3(M, U)$ and output signature $(U, (r + h)d_I)$.
A signature (U, V) can be verified by recomputing h and checking that $e(P, V) = e(sP, U + hH_1(I))$.
This works, because $e(P, V) = e(P, (r + h)d_I) = e(sP, (r + h)H_1(I)) = e(sP, rH_1(I) + hH_1(I)) = e(sP, U + hH_1(I))$. These signatures are called identity based signatures (IBS).

It has been proposed to use IBE signatures for general VANET security [28]. However, this scheme makes very strong assumptions about the available processing power in each vehicle. In aggregation, however, signature verification and other mechanisms that provide guarantees on aggregated messages are not high priority; they can tolerate some delay. Therefore, IBS may be applied to obtain very small public keys at the cost of more delay. Recall that in VANETs, it is assumed that public keys are not known in advance, so certificates should be transmitted to allow verification of each message. An aggregate may be signed by many vehicles, all of which agree with the message; multisignatures allow the combination of all their signatures into a single signature. If IBS can be combined with multisignatures, then it is possible to generate a message M with an arbitrary set of signers L with very limited overhead, because the certificates are only an identity.

5.1.3 Multisignatures and IBS

This section will discuss two schemes; the first illustrates the possibility of multisignatures on an elliptic curve-like group and how they work; the second shows an identity-based multisignature (IBMS) scheme. In the ideal case, identity based multisignatures (IBMS) provide constant signature size and very small public keys, allowing for a large amount of users to sign a single message without prohibitive overhead. The first scheme that will be discussed is based on Gap Diffie-Hellman groups, which is computationally cheap but not identity based. This scheme serves as an example how multisignatures may be implemented. The second scheme will provide IBMS with small keys, constant signature size and the possibility to dynamically merge several (multi)signatures into one multisignature. The core disadvantages of that scheme are that the identity of the signers is not authenticated and the relatively large verification time.

Boldyreva [6] introduced a number of signature schemes based on so called Gap Diffie-Hellman groups. These groups are those where the DDH problem is easy, but the CDH problem is hard. Recall that for supersingular elliptic curves, on which pairings can be defined, this was indeed the case, so such groups exist (see [26] for construction examples). Given such a curve, it is now possible to create the multisignature schemes from [6] in practice. The scheme consists of four phases; setup, key generation, signing and verification.
Setup Select a generator g of group G, set $p = |G|$ and a hash function $H : \{0,1\}^* \to G$.
Key Generation is performed by all participants just like a normal signature scheme; select random $x \in \mathbb{Z}_p^*$ and compute the keypair (x, g^x), where x is the secret key.
Signing of a message M is performed by a subset L of all participants as $H(M)^{x_i}$ for participant P_i. Then the multisignature can be computed by a designated signer D (which can be anyone that knows the parameters of the setup) as $\prod_{j \in L} \sigma_j$.
Verification of signature σ can be done by computing the product of the public keys of all participants $PK_L = \prod_{j \in L} g^{x_j}$ and then using the regular signature verification algorithm to verify the signature, which is simply answering DDH for $g, PK_L, H(M), \sigma$. Note that knowledge of σ, M and L is required to check the signature on M.

[21] introduces an IBMS scheme as a first step to creating an identity based aggregate signature (IBAS) scheme. The scheme is surprisingly simple and signing is a relatively cheap operation; however, verification requires three pairing operations and n point additions (for n participants) and it is not possible to work with multiple CAs in the same multisignature.
Setup Generate two groups G_1, G_2 of prime order q and a pairing $e : G_1 \times G_1 \to G_2$, a generator P of G_1, a master secret key $s \in \mathbb{Z}/q\mathbb{Z}$ and master public key sP, and two hash functions H_1, H_2 that both map text

to G_1.
Key generation Given identity I, compute the keypair with public key $H_1(I)$ and secret key $sH_1(I)$.
Sign Choose random $r_i \in \mathbb{Z}/q\mathbb{Z}$ and compute the signature as $(r_i H_2(m) + sH_1(I), r_i P)$.
Aggregation To aggregate signatures, simply perform point addition for all the individual signatures; $(\sum_i S_i, \sum_i T_i)$ for signatures written as (S_i, T_i).
Verify Given a multisignature (S_n, T_n), verify that $e(S_n, P) = e(T_n, H_2(m))e(sP, \sum_i P_i)$.

Using the IBMS scheme in [21], it is possible to create multisignatures in the desired fashion. What remains to be done for a reliable aggregation mechanism is to build a component that properly generates a fixed message for all vehicles to sign, a way to detect or avoid sybil attacks and a way to prevent a vehicle from tampering with the location and time indication. It may be argued that the use of IBAS is perferable here, but such a scheme requires that all messages be forwarded to obtain a verifiable aggregate signature. The purpose of aggregation in VANETs is to aggregate the messages, not the signatures, so multisignatures are sufficient. In theory, sybil detection by the server is easy; just check which identities belong to which vehicle and see if a duplicate pops up. However, in the current IBMS scheme it is possible for the server to sign messages on behalf of another user; this inherent key escrow should also be addressed.

5.1.4 Aggregate signatures and IBS

This section will discuss the aggregate signatures that [21] introduces. The ideas used by this scheme are almost identical to those in the multisignature setting, except that they need an additional unique message. For similar reasons, it is still not possible to work with multiple CAs in the same multisignature. The scheme works as follows:
Setup Generate two groups G_1, G_2 of prime order q and a pairing $e : G_1 \times G_1 \to G_2$, a generator P of G_1, a master secret key $s \in \mathbb{Z}/q\mathbb{Z}$ and master public key sP, two hash functions H_1, H_2 that both map text to G_1, and a hash function H_3 that maps text to $\mathbb{Z}/q\mathbb{Z}$.
Key generation Given identity I, the CA will compute two keypairs with public keys $P_{i,0} = H_1(I_i, 0)$, $P_{i,1} = H_1(I_i, 1)$ and private keys $sH_1(I, 0)$, $sH_1(I, 1)$ respectively.
Sign Determine a w that has not been used before, which will be the same for each message. It need not be random, only unique to this signing procedure. Compute $H_2(w)$, $c_i = H_3(m_i, I_i, w)$ and generate random $r_i \in \mathbb{Z}/q\mathbb{Z}$; then the signature is $(w, r_i H_2(w) + sP_{i,0} + c_i sP_{i,1}, r_i P)$.
Aggregation To aggregate signatures, simply perform point addition for all the individual signatures; $(w, \sum_i S_i, \sum_i T_i)$ for signatures written as (w, S_i, T_i). It is not allowed to aggregate signatures with different w.
Verify Given a multisignature (w, S_n, T_n), verify that $e(S_n, P) = e(T_n, H_2(w))e(sP, \sum_i P_{i,0} + \sum_i P_{i,1})$.

5.1.5 Computational cost

A distinct disadvantage of using IBE-based schemes is the required computation time. However, similar to ECDSA, recent work has made progress in developing cryptographic co-processors for these groups as well [2, 4], showing that computation times in the millisecond range should be possible when supported by such a co-processor. However, the cost of such a co-processor is currently still unknown; it may be difficult to obtain cheap co-processors with this level of performance.

5.2 Phase 1: Aggregation Phase

The aggregation phase is the phase that finds the locally stable values and sets the aggregation area. Because the aggregation phase controls these two aspects of aggregation, its main security concerns relate to integrity. In addition to the actual cryptographic signature type, this section will discuss privacy issues and message overhead. Although the security mechanisms in this phase are not that different from related secure

aggregation schemes, they introduce a major message overhead; this section will therefore conclude with several optimizations that will be considered for evaluation.

5.2.1 Signatures

Recall that in this phase, the main goal is to find locally stable values, meaning that there are some time constraints to make detection of such values a smooth process. For this reason, identity-based encryption, with its expensive pairing operations, is difficult to justify. Thus, at first glance, it is beneficial to use ECQV ECDSA to generate the necessary signatures, due to the slightly increased efficiency and reduced overhead compared to plain ECDSA. The simplest approach is to use ECQV ECDSA or regular ECDSA depending on what is mandated by standards for regular VANET messages and beacons.

However, the amount of required signatures is too large to transmit in many cases. For this reason, SeDyA will use aggregate signatures to reduce the size. The aggregate signatures will be used on all the FM sketches, which are easy to aggregate, while the LC sketches will retain regular signatures, because the distribution of bits is uniform in this type, meaning that aggregation of signatures will be very rare. However, currently known aggregate signature schemes operate in an identity-based fashion, which reduces the overhead of involved public keys, but increases computational cost. As discussed in Section 5.1.5, the pairing operations that make identity-based encryption so expensive may be computed faster by using a cryptographic co-processor. In addition, note that the use of multisignatures in the finalization phase also requires the use of identity-based encryption, so using it for this phase saves on key material and thus on overhead due to certificates.

5.2.2 Overhead

Another important issue is that of bandwidth overhead. The overhead of signatures and certificates is important, because larger messages and the subsequent wireless channel congestion will also influence the dissemination speed of the messages in the aggregation phase. Unfortunately, the authors that introduced the AM-FM approach [20] do not provide many additional insights on how to limit the overhead. Their evaluation features 256 FM sketches with a size of $\log_2(10^5)$ to count a single binary value, resulting in 4252 signatures. However, in their setting, the event is always binary and there is much more available bandwidth. For VANETs, 4252 signatures is too large by at least an order of magnitude; however, even reducing the amount of sketches by an order of magnitude to obtain 4 sketches of 16 bits each still leaves 64 signatures for a single sketch. In addition, note that more than one of these sets is required. SeDyA requires the inclusion of inflation and deflation free sketches for the amount of participants and the location of the vehicles, in addition to the data, resulting in a total of $8 \cdot 64$ bits of data and a maximum of 512 signatures. Therefore, bandwidth optimization is required.

One very effective bandwidth optimization is the use of aggregate signatures. The simplest method to obtain this result is by replacing the ECQV ECDSA signatures with the IBAS scheme discussed in Section 5.1.4. The required unique w, which is a value that is constant for all signatures that can be aggregated, can be obtained by using the same fixed inputs as used as input for the signing procedure of every bit in the FM sketch, i.e., the fixed point(s) used for location and the time epoch. The additional advantage is that vehicles will also have small public keys, and they will use the same public keys for the aggregation and finalization phase. Aggregate signatures can be applied effectively to FM sketches, where the first 1-bits will likely be set quite quickly. In addition, due to the geometric distribution of bits over an FM sketch, it is very likely that a large sequence of bits will occur at the start of the sketch, compared to a uniform distribution that exists for LC sketches. Since the signatures should never be discarded anyway, and the only important thing for verification is that the receiver knows which certificate was used to sign which bit, the signatures may be aggregated into a single signature. When two aggregates are merged, the aggregate signature that covers the largest amount of bits is used and any further signatures on one bits are aggregated into it, such that the result again covers all the one bits before the first zero bit. Thus, if all the bits in a sketch are one bits, the aggregate signatures will be used to compress the signatures to one signature in each sketch, which will reduce the previous worst-case scenario to $4 \cdot 8 = 32$ signatures. It is possible that up to $512 - 32 = 480$

signatures will be in the sketch (when the first bit in each sketch is zero and the rest is all one), but due to the geometric distribution of values in sketches, this is extremely unlikely. However, the worst-case scenario still contains 512 certificates, assuming each vehicle signs exactly one distinct bit.

One extra way of saving bandwidth can be found in traditional broadcast storm avoidance techniques [46]. SeDyA is implemented with the simplest possible method, which sends a message only with probability $\frac{1}{2}$, if it is generated based on a previously received aggregate. Because this approach is random, it is difficult for the attacker to exploit the technique for its own means, contrary to other broadcast storm avoidance techniques that require more thorough analysis to reach this conclusion. Also, the forwarding probability is sufficiently high, such that the attacker still needs to generate a significant amount of messages to manipulate the behavior of vehicles. This type of pure flooding based denial of service attack cannot be prevented, and this is thus not considered in the evaluation of SeDyA.

Finally, certificates are also an issue; each certificate must be known to the receiver in order to be able to verify it. In the naïve case, certificates are always included, resulting in an additional 512 certificates (worst case) to be included. However, for one-hop message authentication, recent work on certificate omission has shown that significant bandwidth can be saved by intelligently omitting certificates, particularly in a congested scenario [18]. This scenario matches closely with that of aggregation; indeed, most vehicles will be interested to know where on the road the biggest congestions are. In addition, note that vehicles in an aggregation area will repeatedly need the certificates of the other vehicles already, to verify the beacons they receive and to participate in the finalization of an aggregate.

5.2.3 Privacy

Privacy is not a strong concern in this phase, because of its limited duration. However, note that a standard privacy mechanism like pseudonyms may introduce vulnurabilities by allowing the attacker to use multiple key pairs at the same time. This type of sybil attack is usually prevented by restricting pseudonyms to specific time segments (the pseudonym period), which are included in the certificate meta-data. For privacy reasons it may be desireable to allow some overlap in the pseudonym period to avoid linking of two subsequent pseudonyms based on message content [22]. This may be exploited by the attacker, which is why vehicles must always check the contents of a message before contributing to it, using plausibility checks that check the message against the vehicle's current value and the vehicle's neighbour table. However, such checks are always heuristic in nature; therefore, it is necessary for CAs to be able to resolve the pseudonymity provided by pseudonyms in cases of legal dispute or a suspected attack. Although policies for such procedures are out of scope for this thesis, it should be noted that revocation of pseudonymity should be a last-resort measure and the security of a scheme should not rely on it.

5.3 Phase 2: Finalization Phase

Cryptographically speaking, this phase is the most interesting: in the aggregation phase, some manipulation may be unavoidable, but the result of the finalization phase should be as precise as possible. Recall that each vehicle is given the opportunity to submit its signature in this phase, declaring their agreement with a particular aggregate. The protocol defined in the previous chapter collects all these signatures while at the same time computing the multisignature.

5.3.1 Signatures

As already pointed out in the previous section and chapter, this phase uses multisignatures to reduce the overhead. The reason for this is that multisignatures allow a constant size signature that is the product of many signatures, which can be verified using the product of all public keys. This means that the signature size is constant and thus that the overhead is greatly reduced. In addition, by employing the IBMS scheme that was discussed in Section 5.1.3, the certificate size can be reduced significantly to about 20 bytes by retaining only a unique identity and a moment of expiry. However, similar to the aggregate signatures in the previous section, computational efficiency may be an issue. Considering that the finalization phase is

less time-constrained than the aggregation phase, because the distribution of the aggregate will always be a best-effort type of protocol, it is reasonable to take the additional computational effort for granted.

5.3.2 Overhead

The main reason for using IBMS, is to limit the required bandwidth. Recall from the cryptographic background that a certificate for IEEE 1609.2 is currently 117 bytes and is proposed to be extended to 140 bytes, with a signature size of 65 bytes. On the other hand, when ECQV ECDSA is used, one can save an ECDSA signature in the certificate, resulting in a size of about 75 bytes. However, in IBMS, or more generally in IBC, the only required element is the identity, into which much certificate data can be encoded in addition to a globally unique identifier. Thus, a key size of 10 bytes should be more than sufficient, as it is possible to generate 2^{80} different keys this way. Conservatively, consider a key size of 16 bytes; with a current population of around 7 billion people and a pseudonym period of 5 minutes, only 7 bytes are necessary to represent a hundred years worth of unique pseudonyms.[5] The remaining bytes can be used to add certificate meta-data or be randomized for pseudonymity; in practice, the likely solution is that a hash function will be used to hash the identity of the vehicle along with some randomness (different for every pseudonym) to an entire range, after which the validity period will be attached. Thus, it is possible to carry over signatures in a single message (assuming all the data is carried in a separate 802.11 frame). Based on this, it can be concluded that messages will be of reasonable size; however, it remains to be seen whether the amount of distinct aggregates is also reasonable. Regardless, this amount will out-perform SAS, where segments are barely larger than the transmission range (500 meter segments).

5.3.3 Privacy

Instead of employing unique identities, it is possible to simply apply hashing to obtain pseudonyms. However, the time of expiry must remain readable, such as to allow constraints on the validity period of pseudonyms to prevent sybil attacks. The most privacy-sensitive component of SeDyA is the finalization phase; including a signature in this phase will indeed cause the certificate of a vehicle to be disseminated, to a potentially very large region. However, the validity period of pseudonyms is limited to about five minutes; this means that when the map points are located ten kilometers apart and a vehicle drives 120 kilometers per hour, it will need exactly one pseudonym per contribution to an aggregate. Even when a vehicle participates in multiple aggregates, the only revealed information is that a vehicle was on these two segments of road. When this is considered privacy-invasive, the vehicle always has the option not to attach its signature, although this will damage performance of SeDyA.

5.3.4 CA problems

For the multisignature part of the phase, the IBMS scheme discussed in Section 5.1.3 will be used, which inherently supports this method of computing the multisignature, as also noted in the original work. However, because this scheme is based on identity-based signatures (IBS), a few important issues typically associated with identity-based cryptography (IBC) also recur here. Both of these issues relate to CAs: first, in traditional IBC the CA has possession of the private key of every use; second, using multiple CAs in the infrastructure is not inherently possible. Possession of the private key by the CA has been addressed previously by many works, including . While this issue is quite pressing, it is assumed that the CAs are sufficiently trustworthy, delegating the application of more secure IBC to the multisignature scheme as future work. The reason for this is that this approach will require new security proof and thus the introduction of a model, difficult problems and similar concepts from cryptography. The aim of this thesis is to provide security against attackers in the network, not against the compromise of a CA. Second, and for this thesis more relevant, is the issue of using multiple CAs. In a typical world-wide network setting, such as the Internet, it is customary to have multiple CAs for many reasons; it distributes the trust, spreads the computational and storage requirements, provides some redundancy, and for a variety of political reasons. The problem for

[5] $log_2(7 \cdot 10^9 \cdot (100 \cdot 365.25 \cdot 24 \cdot 60)/5) \approx 56$ bits (slightly more, so just over 7 bytes).

IBMS, or perhaps even IBC in general, lies in the fact that the key material of the CA is input for the key material that each vehicle has. If two multisignatures related to distinct CAs are merged, the verification process no longer works for algebraic reasons. The solution to this is simply keeping a separate multisignature for each CA; this solution may seem somewhat cumbersome, but it is fairly elegant as long as the amount of CA remains limited, because signature size is limited. Note that because there is no consensus about the approach and hierarchy of CAs for when VANETs are deloyed, it is difficult to say how often this will happen. Frequently discussed alternatives are either region-based hierarchies or manufacturer-based hierarchies (either vehicle or OBU manufacturers). For SeDyA, the region-based hierarchies are the most optimal setting, because it is likely that most vehicles in a particular area will be from one or two different countries.

5.4 Summary

SeDyA consists of three phases; the aggregation phase, the finalization phase and the dissemination phase.

In the aggregation phase, IBAS will be used in conjunction with the AM-FM technique to aggregate over a region. AM-FM is extended to allow much more dynamic aggregation by using sketches to represent the region in which aggregation is performed. A signature on a bit in the AM-FM means that the signer flipped this bit; it thus proves participation in the aggregation process. However, it does not guarantee agreement with the estimate that the sketch generates, unlike in AM-FM. Privacy will simply be achieved by using pseudonyms, which are restricted to a small segment of time (around five minutes), with little overlap. Inflation and deflation will not work, because vehicles will not add their observations to arbitrary aggregates.

The finalization phase begins when one or more vehicles detect that their neighbors do not agree with the aggregate and decide that they are the border of the aggregation area. These vehicles will initiate the signature collection protocol, which forwards the aggregate along with a multisignature and a set of certificates through the aggregation protocol. In addition, they will provide their location to indicate the edge of the aggregation area. The protocol uses a simple scheme to forward the data through the aggregation area and allow each vehicle the opportunity to sign. Signatures are generated on the aggregated data (the sketches from the aggregation phase, but not the AM) and the location of the finalizing vehicle, using the IBMS scheme from [21] that was discussed in Section 5.1.3.

Finally, the dissemination phase will forward the message in a best-effort way through the network. Although any dissemination scheme may be used, SeDyA would benefit from a scheme where vehicles forward based on freshness, proximity of the aggregation area and possibly the confidence a vehicle has in the message. Heuristics should be deployed to detect attacks as soon as possible. Most notably, to detect remote impersonation attacks, messages that refer to an aggregation area in one direction, but are sent from the opposite direction, are suspicious and should be ignored. Similarly, due to the previously mentioned forwarding scheme, old messages and messages with low confidence are forwarded less quickly than high quality and fresh messages.

Chapter 6

Evaluation

This chapter will discuss the evaluation of the performance of SeDyA. First, metrics are introduced and discussed, followed by a discussion of simulation parameters, a configuration experiment that determines the two thresholds from Chapter 4, the experiments and the results.

6.1 Motivation

The evaluation is divided into three items of interest: first, accuracy will be considered, followed by feasibility and finally security. While security is the main purpose of SeDyA, it is important not to lose sight of both accuracy and feasibility, which are the goals of aggregation. Recall that the main goal of aggregation is to provide a better trade-off between these two aspects, thereby improving the efficiency of the VANET by providing more information for less resources at the cost of some accuracy. Thus, accuracy and feasibility will be measured against each other to determine whether SeDyA still achieves the goals of aggregation, before discussing various metrics for security.

Due to time constraints and limited computational resources, the evaluation will focus on a highway scenario. This scenario is the main scenario for which SeDyA is developed, as well as schemes that have been discussed in Chapter 3. Although SeDyA theoretically provides support for an urban setting, also implemented in the source code, some implementation issues remain, notably the application of aggregation to different driving directions.

The main reference to another scheme for these simulations is the fixed segments approach, which is an adaptation of SAS. However, note that this fixed segments approach uses the aggregate signature approach for its security, instead of symmetric encryption, and it provides additional plausibility checks, taken from SeDyA. These checks ensure that the scheme does not contribute to malicious aggregates, eliminating one potential source of attacks. Removal of this source is essential to provide a good comparison for SeDyA; nothing prevents the implementation of plausibility checks to SAS. In addition, the goal of the simulations includes showing that SeDyA's second phase provides an additional layer of security; by adding plausibility checks to SAS, they are excluded as the cause for improved performance.

6.1.1 Accuracy

Accuracy metrics focus on the quality of the data. Because data quality is very difficult to measure, the accuracy metric below focusses on a particular aggregation setting; determining the average speed. While accuracy of the aggregation mechanism (i.e., the FM sketches) used in SeDyA is by itself not of significant interest to SeDyA, accuracy is the main metric to determine how accurately SeDyA describes a particular aggregation area. In addition, the impact of an attack is mainly described by a reduction of accuracy. Thus, even though the aggregation mechanism itself is not of interest, its impact will be shown in these simulations. Therfore, it should be kept in mind that the accuracy of sketches is considerably limited, as has been shown in the preliminary evaluation of the aggregation phase in Section 4.2.2.

The impact of this inaccuracy is measured in the first experiment, which tests four scenarios: both the fixed segment approach (with a block size of 500 meters) and SeDyA's dynamic approach are tested against a blocked and non-blocked road. The "blocked" roads have a traffic jam, and are referred to as congested; both scenarios can be simulated for a variety of vehicle densities. As noted, the fixed segment approach is similar to SAS. The metric for this experiment is called the *average deviation*, defined as:

$$\frac{\sum_{i \in I} |e - v_i|}{|I|},$$

where e is the average as estimated by the aggregation scheme and v_i is the value for the vehicle i, while I is the set of all vehicles in the aggregation area. The accuracy across the simulation is then determined by taking the average of all these distinct aggregates. The aggregation area is defined by the aggregation mechanism (in the dynamic case), or fixed in advance (in the 500 meter blocks case). The main reason for using the dynamically defined area that SeDyA generates, is that the entire purpose of this area is to provide information about a larger region, while maintaining accuracy. By definition, the dynamically defined area and the accuracy of the data are inter-connected, because the dynamically defined area uses the data to determine its dimensions, while the data is aggregated according to the defined area. The metric abstracts from this inter-connection and determines the accuracy of the whole scheme.

For SeDyA, accuracy can be measured at two points: directly after the aggregation phase, when a vehicle decides to finalize, or after the message has been forwarded through the aggregation area (i.e., at the end of the finalization phase). This allows a detailed analysis of the effect of the finalization phase. In the configuration experiment, the focus is exclusively on the accuracy directly after the aggregation phase; the reason is that the main effect of the thresholds occurs in the first phase.

6.1.2 Feasibility

To verify that each approach is feasible in terms of bandwidth, there are two main approaches: measuring bandwidth directly (bandwidth metric), and measuring the dissemination range of messages (knowledge metric). An example of a knowledge metric would be measuring feasiblity by determining the fraction of the network each vehicle has knowledge about at a given point in time. Measuring bandwidth directly has the advantage that no side-effects from the protocol, such as messages that are ignored due to being incorrect or signatures that cannot be verified, can affect it. This is of vital importance, as the goal of the feasibility analysis is to determine whether SeDyA can run alongside other services in the VANET, not whether SeDyA can fully utilize the bandwidth it is given. The bandwidth metric gives a good indication of the required bandwidth, while a knowledge metric relies on maximum utilization of the available bandwidth. The knowledge metric determines how much information is available to vehicles; very useful information, but only when it is certain that at least the order of magnitude of available bandwidth is configured correctly. Since the bandwidth reserved for SeDyA is unknown, but likely to be limited due to other mechanisms, such as pseudonym updates, revocation protocols and entertainement services, it is better to have a bandwidth estimate than a knowledge estimate that may turn out completely wrong due to the difference in available bandwidth. On the other hand, it is very difficult to account for spatial re-use effects when measuring raw bandwidth usage. However, since measurement using the dissemination range metric requires a very large simulation area to be effective, which in conjunction with the corresponding increase in vehicles creates extremely large simulations, in addition to the disadvantages mentioned above, SeDyA will be evaluated using the bandwidth metric. Bandwidth usage will be measured by determining the exact size of SeDyA messages, excluding overhead from the IP and 802.11p layers, as well as any additional constraints on the physical layer. Since SeDyA is supposed to run in the sevice channel in 802.11p, bandwidth consumption of beaconing mechanisms is ignored for all these experiments, although it may have some impact.

6.1.3 Security

SeDyA features security mechanisms in two phases, as opposed to just one in schemes from related work, like SAS. The first security experiment will focus on the first security mechanism that SeDyA has in common with

related work, by showing that attacks against this mechanism exist. Just like for the definition of correctness of an aggregate, a major challenge is when to consider a produced aggregate to be legitimate, and how to derive a definition for the security of such a scheme. The main reason for this difficulty is due to the fact that inaccuracy is inherent to aggregation; therefore, one cannot define a metric that considers an aggregate with an error as an attack, because such an error may be inherent to aggregation. In addition, the impact of an attack is much more relevant to the aggregation mechanism than the frequency of attack success, given that such a frequency would inherently be high. Consider for example that an attacker manages to influence the speed value contained in an aggregate by 0.0001%; this could be considered an attack, but with an error due to aggregation that is (for example) 1%, this attack is completely meaningless. For this reason, attackers are considered as an additional source of inaccuracy, in addition to the aggregation mechanism and the plausibility checking mechanisms. This allows to show the overall impact on the quality of the data for a varied number of attackers that operate independently, but with a similar goal. Therefore, for both the security experiments, a number of attackers have been implemented that cover the most general approaches an attacker can take: attacking the value and attacking the region. In the first security experiment, which focusses on the first phase, only the attacker that combines all the naïve attacks into a concentrated attack is considered. The other attackers were implemented, but since the Smart attacker combines their approaches and exploits knowledge of SeDyA's thresholds, this attacker has the focus. More precisely, this Smart attacker exploits the fact that an attacker can use a received sketch and modify it, rather than using a predefined scheme to determine a target value (i.e., the naïve attacks that were implemented but not used). As previously discussed in Section 3.4.1, such an attack on FM sketches is executed by flipping the first 0-bit in each sketch, causing the estimate of the sketch to increase. The same is applied to the LC sketches that SeDyA uses.

As a second security experiment, the added value of SeDyA's finalization phase will be tested. This experiment will evaluate the signing procedure that SeDyA provides. The expectation of this signing procedure is that a relatively large fraction of the participating vehicles will sign a legitimate message, while an attacker's message should have a limited amount of signers. In the ideal case, vehicles that are in the finalization phase will drop attacker messages; however, due to the high inaccuracy of the sketches, this is a solution that may exclude many legitimate messages, potentially enabling denial of service attacks. To analyze the results, a number of simulations with and without attackers are performed. The fraction of signers is graphed for attacker and non-attacker cases; the difference between these two graphs determines how well SeDyA's second phase works, as a higher difference makes it easy for a vehicle to distinguish benign and malicious messages. Again, several attackers are implemented; for simplicity, and to obtain results that are as widely generalizable as possible, each of these scenarios assumes the attacker has full control over the final message produced in the aggregation phase. For this experiment, the attackers are an Area attacker and an Inflation attacker; the Inflation attacker operates analogously to the Smart attacker in the previous security experiment. The Area attacker aggressively attempts to change the aggregate by configuring the finalization area to be very far from her actual location. Both attackers assume control of the finalization process by declaring themselves as finalizing vehicle. The Area attacker exploits the fact that the second phase stores the location of the finalizing vehicle, which is the attacker. The Inflation attacker has complete control over the message from the first phase, and will use a high value as the estimate; the Deflation attacker is equivalent, but uses a low value as estimate.

6.2 JiST/SWANS

To perform the experiments described in this chapter, SeDyA has been implemented in JiST/SWANS. JiST/SWANS (Java in Simulation Time / Scalable Wireless Ad hoc Network Simulator) is a discrete event-based network simulator, developed between 2002 and 2005 by Barr [3] at Cornell University. It has since been improved by the Universities Ulm and Twente with enhancements to allow the simulation of VANETs, including an implementation of 802.11p, the STRAW mobility model, beaconing applications, security applications and various VANET protocols. Some of this source has been released as a GPL patch for the

original JiST/SWANS distribution in 2008 [48], while other code remains internal so far[1].

6.2.1 Implementation of SeDyA

SeDyA has been implemented largely as described in Chapters 4 and 5, including the three phases and most of the message exchanges. In addition, SeDyA uses the existing secure beaconing code, which provides security for beacon messages, and an identity generator that simulates the use of pseudonyms. As discussed in Chapter 5, two threshold values are introduced in SeDyA; an agreement threshold that determines when a message still agrees with the data of the vehicle and an edge threshold, which decides when a vehicle is at the edge of an aggregation area. Note that the computations that involve the edge of the aggregation area involve the difference in the neighbor table: the vehicle compares the average for the vehicles behind and in front of it, while the aggregation threshold compares a received aggregate with the vehicles individual data and the average of its neighbor table. Other checks to verify packets include mainly the verification of signatures and checks that determine freshness of a message. The physicial model checks are not implemented, because they involve a lot of domain-specific knowledge and most of these failures would not pass the check against the neighbor table anyway. Cryptography itself is not simulated; the value assigned to each public key by the vehicle's identity generator is also used to determine whether a signature is still verifiable. This saves a lot of computational resources without impacting the accuracy of the simulations. For multisignatures and aggregate signatures, the same approach is used, but the public keys are summed, as are the signatures. For the LC Sketches, simulations crashed due to memory usage when using aggregate signatures; thus, the aggregation phase uses regular signatures for this sketch type.

Countrary to the real world, JiST/SWANS currently provides only a single channel for communication using when using 802.11p. In reality, standards have defined multiple channels that are to be used for different purposes. Those relevant for SeDyA are the 6 Mbit/s control channel, for beacon messages, and the (up to) 27 Mbit/s services channel, for SeDyA's messages. In the simulations, all message are sent on the control channel, which is the only available channel. The reason for this is two-fold; first, it is important to study bandwidth consumption in a worst-case setting, to be sure that SeDyA will work. Second, the services channel should not be fully dedicated to SeDyA; other services will need bandwidth to operate as well. Finally, it should be noted that SeDyA considers secure beaconing in its analysis as well, with a fixed size of 209 bytes, including security payload. See Table 6.1 for a full overview of the important simulation parameters.

6.2.2 Simulation parameters

To perform consistent simulations, most parameters for the simulator are fixed. The discussion of each experiment describes the relevant variables that are varied, typically including the amount of vehicles, size of the area and inclusion of attackers. The mobility model for the simulations is a simple car-following model on the highway scenario, which includes overtaking and collision avoidance behavior. The urban scenario was not rigorously simulated due to some implementation issues; however, initial simulations indicated that SeDyA should be bound to a specific road, rather than a region of roads.

The beaconing rate is set to 10Hz, which is a common and standardized setting; beacons include location information, heading and driving speed, in addition to the security mechanism. SeDyA does not interact with the beaconing mechanism, except for using the produced data; separate key mechanisms are used.

6.3 Optimizing SeDyA

Due to the large amount of parameters, this section will first configure the two thresholds for SeDyA in such a way that they are optimal when considering both accuracy and bandwidth efficiency. Both of these parameters will then be fixed for the remainder of the evaluation. The thresholds are configured for accuracy

[1]Due to all these licensing constraints for the base code, there will not be a public code package for JiST/SWANS. The author can be contacted for specific inquiries.

Group	Parameter	Setting
	Duration	200 seconds
Mobility model	Scenario	Highway
	Model	JiST/SWANS Ulm Highway model[a]
	Lanes	3
	Lane Length	5000 meter
	Maximum speed	30 m/s
	Maximum acceleration	1 m/s (5 m/s breaking)
	Average speed	18 m/s
Physical layer	Fading	RayLeigh
	Path loss	TwoRay
	Transmit power	10.9 dB
	Noise model	Additive
MAC layer	MAC	802.11p
Network layer	IP implementation	JiST/SWANS implementation[b]
	Maximum packet size	4096 bytes
Beaconing	Interval	10 Hz
	Beacon size	209 bytes
SeDyA	Vehicle counter sketch size	64 bits
	Location sketch size	8 bits per sketch with 4 sketches for PCSA
	Payload sketch size	8 bits per sketch with 8 sketches for PCSA
	Hash method	MD5 (and SHA1 for PCSA)

[a]This includes a simple car following model, collision avoidance, lane changing and overtaking.

[b]This implementation is like regular IP, but does not support fragmentation. Thus, packages larger than the maximum size are dropped.

Table 6.1: The simulation settings common for all simulations.

and bandwidth efficiency because SeDyA must first and foremost effectively perform the aggregation process, before security is considered. The reason for this is that while security is the goal of SeDyA, efficiency and accuracy are the main purpose of aggregation mechanisms. Therefore, is required that these are achieved before security is even relevant; a more secure aggregation mechanism that is inefficient and has poor accuracy will (almost) never be used[2].

6.3.1 Configuration and parameters

The accuracy and bandwidth metric discussed in Section 6.1 are used for the evaluation; the edge and aggregation thresholds are varied between 0.5 to 3.5 with steps of 1 and between 2 and 10 with steps of 2, respectively. This range was selected for further investigation after performing initial experiments. In addition, a very high agreement threshold (16) was included to observe the effect of an extremely high aggregation threshold. Note also that an edge threshold of 0 would imply that each vehicle is an edge node and an agreeement threshold of 0 would imply that vehicles should never agree with deviating values, no matter how close. The agreement threshold value of 2 means a deviation of 2 meters per second is the upper bound for the vehicle to accept the message as legitimate, and so on; for very high values of the agreement threshold, almost every message is aggregated. Considering that in Section 4.2.2, a relative error of over 0.5 was quite common, and the maximum speed is 80 kilometers per hour (about 22 meters per second), the minimum agreement threshold of 2 should be a good minimum. It was hypothesized that for higher values of the agreement threshold, accuracy would become a problem quite quickly; the choice of 10 should be more than sufficient to show this, or at least decide whether simulations with higher values are necessary. The reason for the difference in the values is that the edge threshold compares two averages (subsets of the neighbor table), while the agreement threshold compares against both the neighbor tabe and the vehicle's current value. In addition, the inaccuracy due to aggregation is also involved in the agreement threshold, which is not the case for the edge threshold.

Simulations were performed with 75 and 250 vehicles on a 5 kilometer stretch of road, using a non-congested setting and a fully congested setting. Higher vehicle densities were not considered due to relatively high memory consumption of simulations. The amount of repetitions was limited to three for this configuration experiment, as the goal is to get an idea of the effect of the thresholds. Other parameters have been set in the same way as discussed in the simulation parameters section.

6.3.2 Results

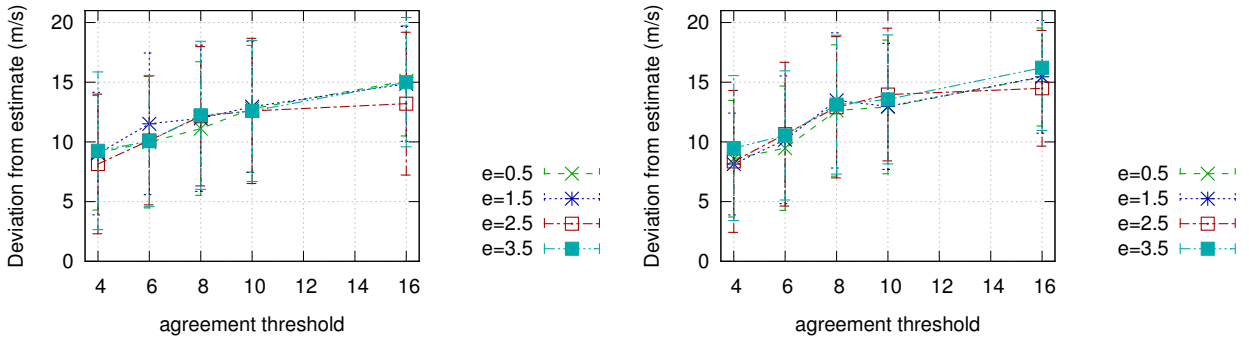
Before proceeding to the results of the analysis, it should be noted that all the simulated data is made public[3]

Figure 6.1 shows the influence of the agreement threshold on the accuracy metric, for both low and high density settings (as noted, 75 and 250 vehicles on 5 kilometers of 3 lane highway). The different graphs, denoted by a value for e, represent different settings for the edge threshold. As expected, this graph shows that a higher agreement threshold reduces accuracy (almost) linearly. This makes sense: if vehicles are more lenient towards estimates, the difference between the average and the vehicle's value will be higher. Another expected result is that the accuracy for lower density regions is quite bad, as seen in Figures 6.1b and 6.1a; this is due to the high variations in speed that the mobility model provides.

The impact of the different edge threshold settings is lower than expected; the hypothesis was that a higher edge threshold would lead also to a loss in accuracy, because more vehicles would contribute to the same aggregate. Instead, the graphs show that all the differences are well within the bounds of the standard deviations for the uncongested setting and congested low density setting (Figures 6.1a, 6.1b and 6.1b), while in the congested high density setting (Figure 6.1d), the higher edge threshold performs slightly better. The small difference can be explained by noting that the inaccuracy of sketches is relatively high, so the area

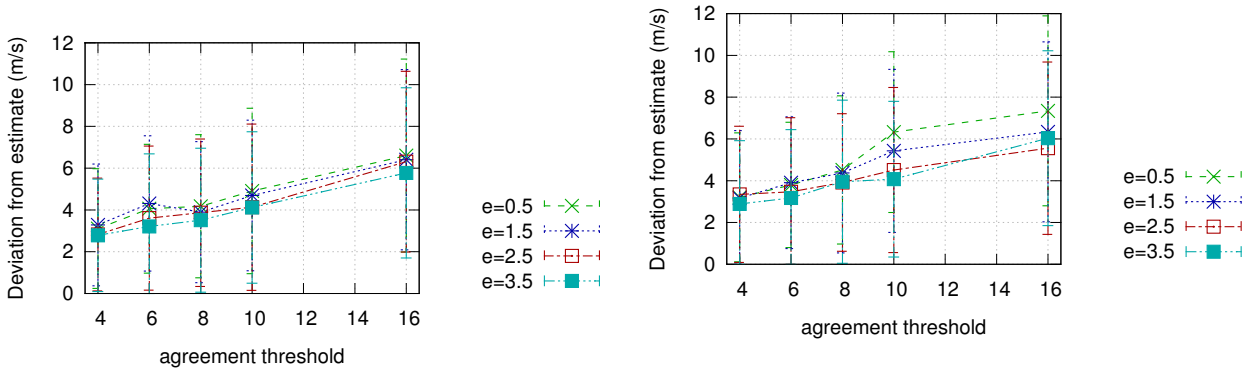
[2]For security mechanisms in general, this is not the case; however, applications that use aggregation will already be fault-tolerant. Note also that aggregation is a supplementary application; its functioning is not required for VANETs to function, so not having aggregation may be better than secure but poor aggregation.

[3]All the data will be published in an organized fashion here: `http://91.121.197.132/work/master_thesis/baf75cb3f019a9528553d9c535ff765b8c8cd40a/data/`. The current experiment is preliminary and thus labeled with 'experiment 0'.



(a) Low density: 15 vehicles per kilometer, no congestion

(b) Low density: 15 vehicles per kilometer, with congestion



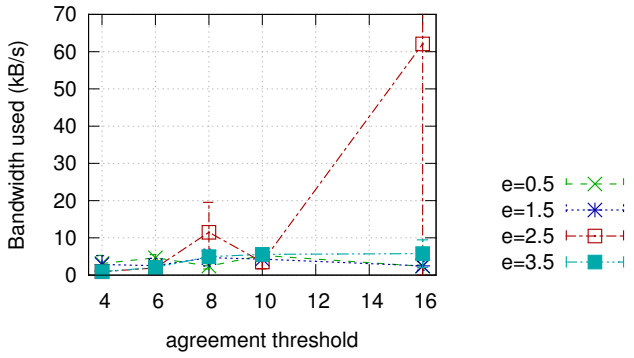
(c) High density: 50 vehicles per kilometer, no congestion

(d) High density: 50 vehicles per kilometer, with congestion

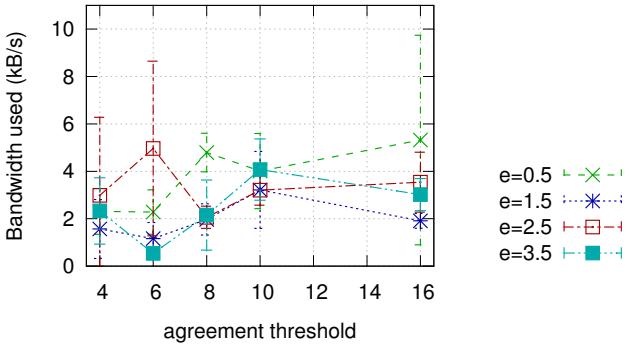
Figure 6.1: Influence of thresholds on accuracy.

described by SeDyA is likely to be larger than necessary for the aggregates produced by lower thresholds. These aggregates are typically very small; they consist of data from less than five vehicles, so the locations of the contributing vehicles are very close together. Thus, because the edge of the area is one of these five, even a small error will cause the area to grow quickly; the relatively high errors for sketches cause this to include many extra vehicles. In the congested setting, the speeds vary more over the road, and thus the impact of this larger area is shown by a somewhat larger value for smaller edge thresholds.

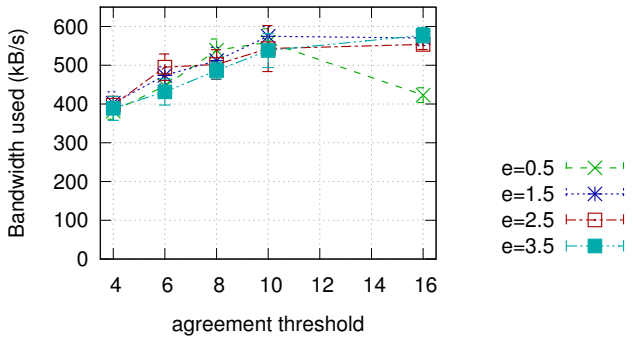
Next, Figure 6.2 shows the impact of the same settings on overall bandwidth usage. As noted, this bandwidth usage is only comprised of SeDyA's messages and their payload, not the additional overhead from other network layers or retransmissions. Furthermore, the bandwidth usage is that of all phases combined, rather than the usage of one specific phase. This graph shows that the cost of security is much higher than the bandwidth saved by using large but highly aggregated messages, as set by a higher agreement threshold. This is especially the case in the high density scenarios, where many vehicles sign the messages they see in phase 2. The low density scenario shows some variations in the congested scenario; this is due to the fact that congestions in a low density scenario dissolve quickly, leading to highly variable conditions. In addition, this scenario and the



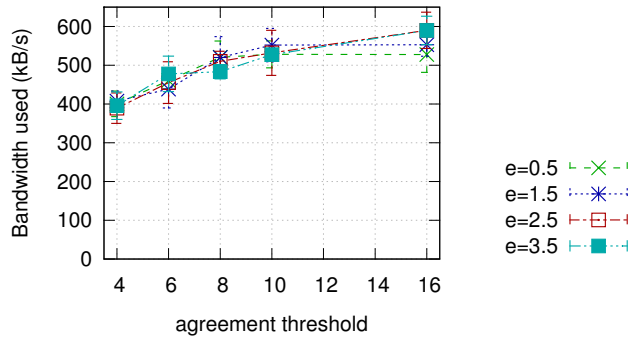
(a) Low density: 15 vehicles per kilometer, no congestion



(b) Low density: 15 vehicles per kilometer, with congestion



(c) High density: 50 vehicles per kilometer, no congestion



(d) High density: 50 vehicles per kilometer, with congestion

Figure 6.2: Influence of thresholds on overall bandwidth usage. Standard deviation in Figure 6.2a is cut off on both edges of the graph due to its size.

low density scenario have a single result with a very high value, which causes the spikes in the graph for the edge threshold of 2.5. Finally, the scenario without congestion and a high density has a lower low value for edge threshold 0.5 and agreement threshold 16. The most likely explanation is that this threshold causes so many collisions, due to the high amount of messages as well as the high amount of signatures, that messages are lost. Note that SeDyA does not provide any re-transmission mechanism in phase 2: if a message is lost, it will not generate any additional traffic.

It should be noted that it is possible to reduce the overhead further by introducing signing policies in the finalization phase, such a policy could define that not everyone should sign, but only a fraction (for example, sign with probability 0.5). Another possibility is to adapt a more efficient broadcast mechanism to collect the signatures; the current mechanism essentially re-broadcasts a fairly large message $O(n)$ times; once for every vehicle in the aggregation area. However, due to the potential security implications caused by limiting the amount of messages or signatures, these options are not further investigated. For example, against a contention-based mechanism to limit the amount of messages, the attacker can perform a selective jamming attack to reduce the amount of signatures generated on a message, by jamming certain messages to cause additional loss. While such an attack can be prevented by using a vehicle's neighbour table, this puts various attacks on the neighbour table in scope; this type of optimization is thus left for future work.

As a conclusion, the thresholds are configured to be 4 for the agreement threshold and 3.5 for the edge threshold. This configuration has shown a good balance between accuracy and bandwidth, considering that the bandwidth usage is significant in all simulations.

6.4 Simulation Results

6.4.1 Accuracy

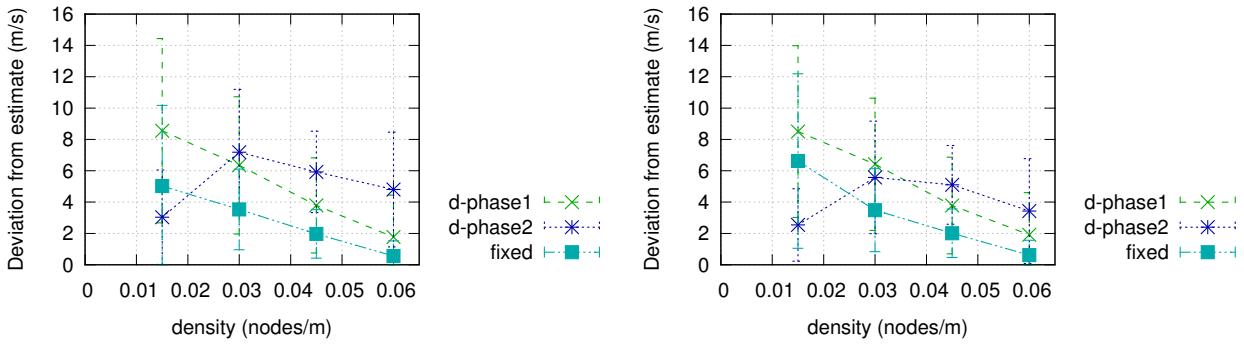
The accuracy depends on a few newly introduced mechanisms, notably including the dynamic aggregation area and the efficiency of the dissemination mechanism. Thus, it will be measured for different scenarios to determine the performance of SeDyA, using the previously discussed metric. The experiment will run SeDyA and the fixed segments approach to identify the accuracy of each approach as per this metric. In addition, the impact of SeDyA's finalization phase will be analyzed, by measuring the accuracy both at the start and the end of this phase. The main variable in these simulations is the density of the network, set by the total amount of vehicles simulated on the 5 kilometer stretch of road; the amount of vehicles is varied from 75 to 300 in steps of 75. In addition, the scenario is varied between congested and uncongested; the speed of vehicles in the uncongested scenario are more variable over time.

The results are shown in Figure 6.3. First, observe that the result for accuracy measurements after the finalization phase ('d-phase1' in the figure) is not as good as that after the finalization phase ('d-phase2' in the figure), especially for lower densities. The reason that high densities have somewhat better values for 'd-phase1' is due to the higher density (and therefore more consistent speed), coupled with the fact that in phase 2, collisions cause not all of the high quality aggregates to reach the end of the area. In fact, more collisions will occur surrounding high quality aggregates, because the protocol requires each vehicle to transmit its signature on the message, if it agrees with the payload.

Second, the result clearly shows that the fixed segment approach ('fixed' by in the figure) is the superior value, at least for higher vehicle densities. However, this is to be expected; recall that the fixed segment approach divides the road into fixed parts of about 500 meters in length; such a segment is roughly contained within the communication range of a vehicle, if the vehicle is placed at the center. Therefore, if such a vehicle sends a message, it will immediately exit the aggregation area and therefore transition to dissemination (because phase 2 does not exist in the fixed segment approach). Although the aggregation mechanism has a minimum duration implemented to avoid this problem, the core issue remains; the aggregate consists of only the sensor data of very few vehicles, which consequently causes a higher accuracy. However, as density increases, and available bandwidth per vehicle decreases, the deviation in SeDyA's aggregates goes down as well. Apart from the above reasons, it should also be noted that the higher density scenario is simply more stable in this highway mobility model; free-roaming cars have widely varying speeds. Finally, it should be

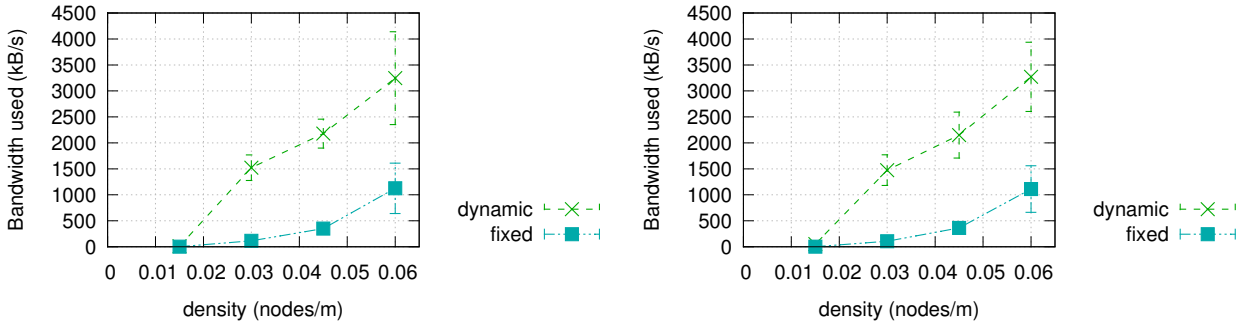
noted that the inaccuracies for the fixed segment approach mainly come from the FM sketches, which can introduce significant errors as discussed in Section 4.2.2.

In conclusion, the improvement in accuracy is not as significant as expected, for high density scenarios, but the second phase definitely achieves the goal it was designed for. To know whether SeDyA is actually useful, one must also consider bandwidth usage and security, which will be done in the following sections.



(a) No congestion (b) With congestion

Figure 6.3: Influence of density and phase 2 on accuracy.

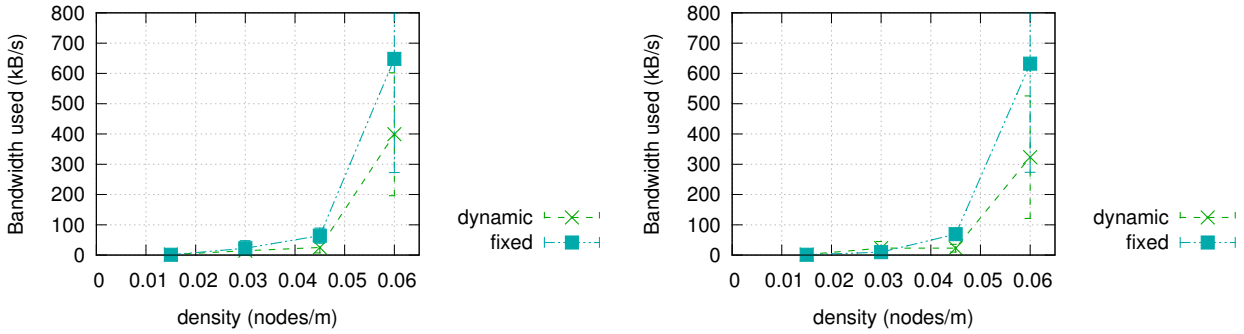


(a) No congestion (b) With congestion

Figure 6.4: Comparison of overall bandwidth usage.

6.4.2 Feasibility

Now that it has been established that SeDyA is sufficiently accurate, it is also important to consider overhead; the bandwidth consumption of the simulations in the previous section is now examined. Recall that bandwidth is measured by the size of the SeDyA messages that are transmitted, excluding overhead from the lower network layers. As shown in Figure 6.4, as well as the configuration experiment, the total bandwidth



(a) No congestion

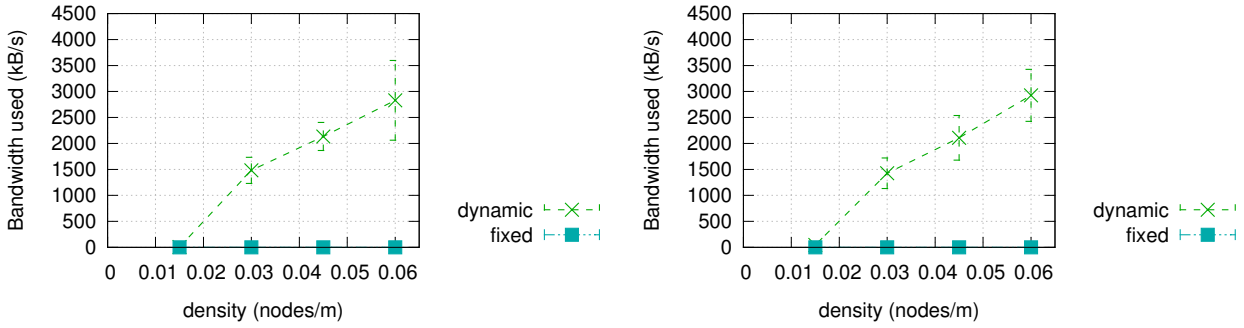
(b) With congestion

Figure 6.5: Comparison of bandwidth usage in the aggregation phase. Standard deviations are cut off in both figures to show some detail for lower densities.

usage of SeDyA is quite high; however, if one looks at only the bandwidth consumption in phase 1, then it can be seen that SeDyA does indeed perform the task it was designed for. More precisely, the dynamic configuration of the area does indeed have the intended effect that less overhead is produced in the first and third phase (shown in Figures 6.7 and 6.5, respectively). Recall that in the first and third phases, aggregation and dissemination are performed, respectively; it is therefore also the main bandwidth usage source that needs to be reduced to conclude that more aggregation is positive for bandwidth usage. This shows that the dynamic component of SeDyA has its desired effect, as one could expect from other dynamic aggregation mechanisms. However, the additional security that SeDyA aims to provide through the finalization phase is quite expensive in terms of bandwidth, as shown in Figure 6.6: the vast majority of the resources are used by this phase. This figure has 0 bandwidth for all fixed segment cases, because a finalization phase does not exist there. Consequently, as can be observed in the overall bandwidth usage, the total cost of SeDyA is signficant when compared to the fixed segment approach; this suggests that a trade-off may be the best achievable result. This trade-off provides better accuracy in lower densities, and the higher level of security that SeDyA aims to provide, in exchange for more bandwidth overhead. To learn more about this trade-off, the higher level of security first needs to be confirmed.

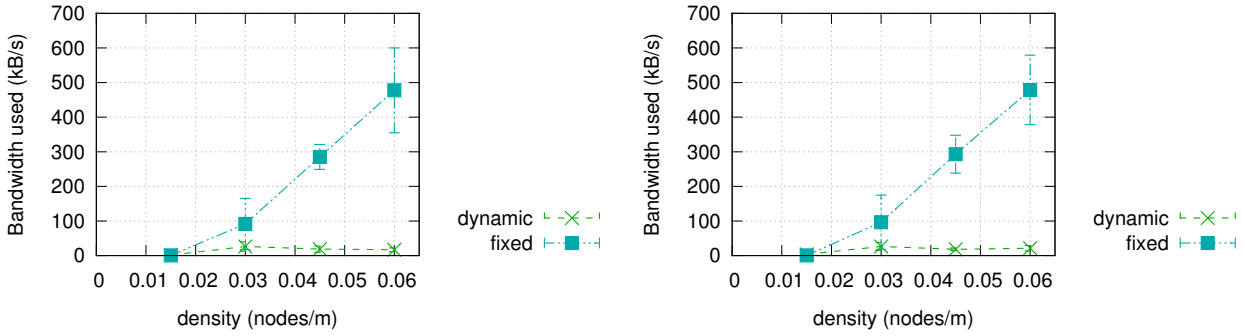
6.4.3 Security

Finally and most importantly, SeDyA's security must be analyzed. In this section, the attacker model from Section 2.1.2 will be used to evaluate how well each phase protects against the relevant attacks. However, note that sybil attacks have already been excluded by assuming a limited amount of pseudonyms that may be active. First, SeDyA is compared to the fixed segment approach, using attacks on the first phase, designed to have maximum impact. This attacker is designed specifically to have the greatest possible impact without being discarded by SeDyA; the same attacks are applied to the fixed segment approach to simplify the implementation and make the implementation of the plausibility checks consistent. Although the attacker is primarily designed to attack SeDyA, the vulnurability she expliots lies in the usage of FM sketches, and the attack can thus also be exploited in the fixed segment approach. Subsequently, potential new attacks that can be performed on phase 2 are analyzed to confirm that this phase does not introduce new vulnurabilities. Since these attacks are specifically against SeDyA, only SeDyA is considered in that analysis.



(a) No congestion (b) With congestion

Figure 6.6: Comparison of bandwidth usage in the finalization phase.



(a) No congestion (b) With congestion

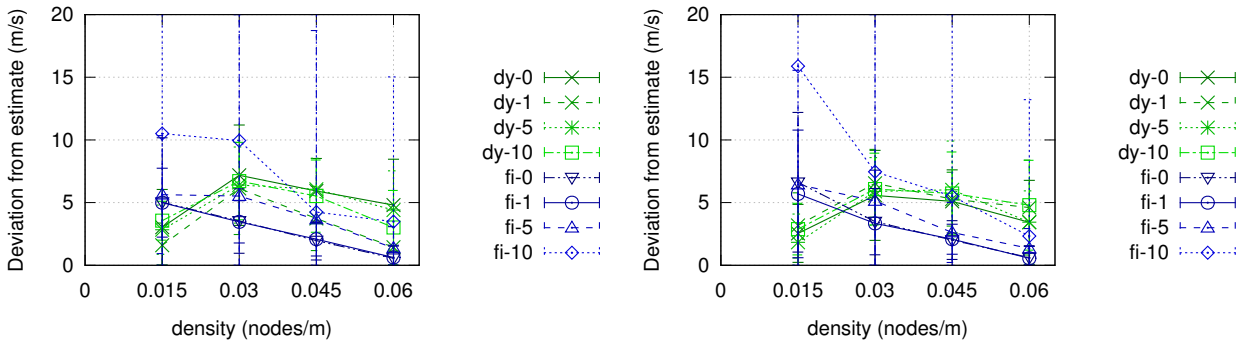
Figure 6.7: Comparison of bandwidth usage in the dissemination phase.

Attacking phase 1

As discussed previously, the attack itself is performed by flipping the first zero bit in each sketch, demonstrating an inflation attack. Recall that the first zero bit is exactly that bit that is used to determine the value that the sketch represents. Flipping this bit may occur legitimately, so there is no way for any subsequent receiver to determine whether the attacker tampered with the message or simply contributed legitimately, although it may be possible to mark some of these messages as suspicious. For example, if a vehicle overhears two messages, where one message is exactly the same as the others, except for exactly those bits an attacker would choose to flip, the message may be marked as suspicious, because this is a very rare occurence. More generally, SeDyA and the fixed segment approach can attempt to detect this type of attack by using a deviation check (i.e., the agreement threshold from the configuration experiment). The impact of an attack is measured by the deviation in the aggregate; in this experiment, the average deviation over all messages will be used to analyze the effect of the amount of attackers on SeDyA and the fixed segment approach for different vehicle densities. Figure 6.8 shows the results for the smart attacker; the amount of attackers is either 1, 5 or 10. Note that 10 attackers corresponds exactly to one attacker per segment for the fixed segment approach, as the simulations run on a 5km stretch of road with segments of 500m. The attackers do not collude, but they use the same algorithm towards the same goal: incrementing the value in the sketch. Other than this manipulation in phase 1, the attackers are entirely honest.

As shown in the figures, the impact of a single attacker on the overall data quality is insignificant in both cases; however, as the amount of attackers grows, the impact on the fixed segment approach increases drastically, while for SeDyA it remains the same. In addition, note that the fixed segment approach shows a large standard deviation (for all but the highest density it goes off the chart); this indicates that the impact of some attacks is very large indeed. SeDyA performs well in low density scenarios, as well as scenarios with many attackers, while as the vehicle density increases, the fixed segment approach improves slightly, in exchange for a higher stanard deviation. However, due to the lower variation, using SeDyA in a general setting is desirable from a security perspective. Recall again that the metric shows the impact averaged over all messages; the high standard deviation for fixed segments may imply that the attacked messages have a high impact, but there is sufficient redundancy to (on average) compensate for this impact. In addition, as an interesting side-effect, some attacks on SeDyA cause errors due to sketches to be excluded, resulting in a better accuracy than the setting without attacks. This is particularly the case in scenarios where the difference between the values of vehicles is relatively high, i.e., when there is no congestion, as shown in Figure 6.8a. Finally, SeDyA also provides a list of identities that should help determine the accuracy of a message, while the fixed segment approach does not provide this functionality. For this analysis, however, the payload is not considered; the analysis focusses on all received messages, so only messages that are dropped by SeDyA's second phase for their inaccuracy are not included here. The reason for this choice is that the messages that arrive at the end of the aggregation area have the support of at least so many vehicles as are required to disseminate the message through the aggregation area, as vehicles do not forward messages they do not agree with. Because some scenarios require that even uncertain messages be accepted, and due to the added complexity of another threshold mechanism that would be required to decide whether to accept the message, the analysis does not consider this part.

In conclusion, this experiment has shown that SeDyA signficantly reduces the standard deviation of the overall data quality, when compared to the fixed segment approach. This is especially notable in scenarios with a significant amount of attackers; lastly, it has been shown that a single attacker in her own segment does not have a large impact on the overall data quality of the aggregates that are produced by the network. SeDyA therefore offers an alternative to the trade-off between the lower level of security, but also lower bandwidth usage that the fixed segment approach offers. At this point it should be noted that SeDyA can be adapted to provide a trade-off that can be tweaked: either by defining more restrictive signing policies or by defining a better protocol for attaching signatures. In addition, SeDyA's security mechanisms in the aggregation phase may be replaced entirely by the plausibility checks to free up bandwidth. These three mechanisms can allow SeDyA to trade security for more bandwidth efficiency; future work is needed to make a rigorous analysis of the impact of these changes.



(a) No congestion

(b) With congestion

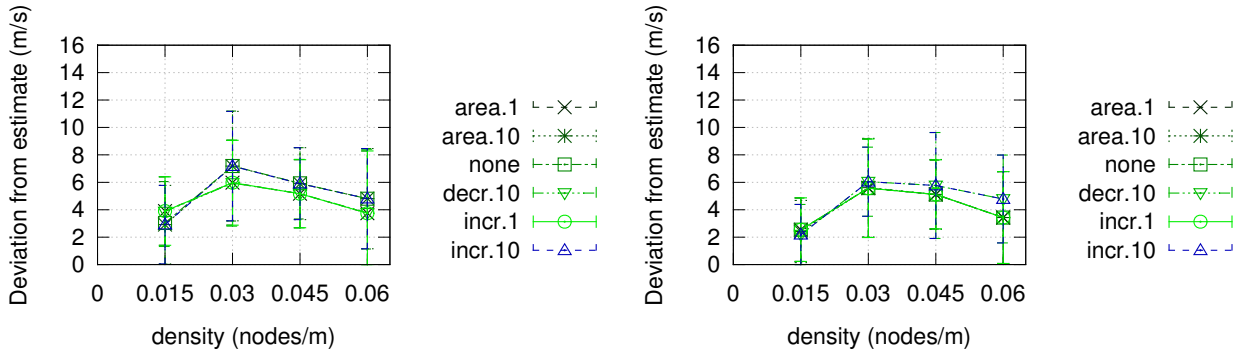
Figure 6.8: Influence of vehicle density and attacks on the first phase, as measured by accuracy after the second phase, for fixed (fi) and dynamic (dy) approaches. The number indicates the amount of attackers. Standard deviations for the fixed approach have been cut off to show more detail.

Attacking phase 2

In this last experiment, it is analyzed whether SeDyA introduces significant new vulnurabilities that may be exploited by the attacker. Since the algorithm used for the multi-signature scheme is secure, the best an attacker can to attack this component is perform a denial of service attack, by for instance, adding garbage to a signature, identity or the message. However, this is a capability that always exists, irrespective of the algorithm used, because the security mechanism cannot control the messages an attacker sends. Similar reasoning applies to the attacker dropping messages that should not be dropped. The remainder of this section will evaluate the attacks that the attacker can perform only on SeDyA, by modifying the messages in the aggregation phase, or by controlling the input of the finalization phase.

Note that the attacks on the aggregation phase are exactly those that were covered in the previous section (and were adapted to also work on the fixed segment approach). Thus, what remains to be examined are the attacks that are possible when the attacker is in a position to finalize a message. In such a case, the attacker determines which message is the final message; thus, SeDyA should prevent these messages from being forwarded, or cause them to have few signers. For this attack type, several implementations exist, as previously discussed. The ones tested here include the standard incrementing attacker (denoted `incr`), which simply configures a higher value, and the area attacker (denoted `area`), which modifies the area by configuring a different position. The decrementing attacker is analogous to the incrementing attacker, but reduces the value (denoted `decr`).

Figure 6.9 shows the results; the impact of the attacks is almost negligible: all points in the figure are will within each others' standard deviations. Note also that some of the attacker points are actually slightly below the line without attackers, in the case without congestion. This is similar to what was observed in attacks on the aggregation phase; again, the reason for this is that the attacker is intelligently attempting modification, incrementing with just half the agreement threshold. This causes sketches with only slightly too high values to be discarded due to the additional error introduced by the attacker. From these results, it is concluded that SeDyA does not introduce new vulnurabilities that are not covered by the finalization phase.



(a) No congestion

(b) With congestion

Figure 6.9: Influence of vehicle density and attacks on the second phase, as measured by accuracy after the second phase, for different attacker types; decr is the decrementing attacker, incr is the incrementing attacker, area is the area attacker. The number indicates the amount of attackers.

6.5 Summary

This chapter has rigorously tested SeDyA's capabilities through simulation, examining three main metrics: accuracy, feasibility and security. Before putting SeDyA itself to the test, SeDyA was configured by testing different values for its edge and agreement thresholds, which control the degree of aggregation and the lenience of vehicles towards measurement errors, respectively. From these results, an edge threshold of 3.5 with an agreement threshold of 4 was selected as the best combination. Subsequently, SeDyA has been compared against an implementation of the fixed segments approach. To exclude influences from the plausibility checks that SeDyA uses, the same checks have been added to the fixed segment approach. The fixed segment approach itself is an adapted version of SAS [24], using aggregate signatures, as suggested in the original paper.

The core result of the analysis confirms the intuition that there is a trade-off between the bandwidth usage, the accuracy of an aggregation mechanism and the resilience against attacks. Before SeDyA was introduced, three general approaches existed: either forwarding every single beacon, secure aggregation and normal aggregation. Recall that forwarding every single beacon provides the highest possible level of security, but also the worst bandwidth performance. On the other hand, normal aggregation mechanisms, without security, are very bandwidth-efficient but allow the attacker full control. As discussed in Chapter 3, some existing work provides secure aggregation, aiming to combine both into a good level of security with reasonable bandwidth usage. While these approaches succeed to some extent, some potential attacks still remained, as identified in the sections that describes these approaches. SeDyA attempts to fill the gap between the related work and the highest possible level of security, but at a better level of bandwidth usage, compared to forwarding all beacons.

The analysis has shown that SeDyA is less bandwidth efficient than related work, but provides higher accuracy in low density scenarios and stronger security guarantees, especially in the presence of multiple non-colluding[4] attackers. The reduction in bandwidth efficiency is quite significant, but the current implementation of SeDyA provides the highest possible level of security that SeDyA can provide. In addition, it has been demonstrated that SeDyA's security mechanisms do not provide a surface for new attacks, by giving the attacker full control over the message that SeDyA provides as aggregate.

[4]However, the attackers do amplify each other, because they have the same goal.

SeDyA's significant bandwidth reduction may be compensated in future work by analyzing the possibility of optimizations in its protocols. Specifically, SeDyA's finalization phase consumes a large amount of bandwidth for the security it offers. This finalization phase may be optimized by implementing signing policies, which limit the amount of signatures produced in this phase, by implementing a more intelligent method to iteratively contruct the multisignature, or by completely discarding the security mechanisms in phase 1. This last solution is of particular interest, as it has been shown that the impact of an attacker with full control over the input of the finalization phase is very low.

Chapter 7

Conclusion

This master thesis has introduced and discussed the challenges surrounding secure aggregation, VANETs and dynamic aggregation, shown current approaches in related work and used the results of these discussions to design a new scheme. This new scheme, Secure Dynamic Aggregation (SeDyA), provides dynamic aggregation, including the desired level of security. In addition, it has been shown that goal of dynamic aggregation is preserved; however, the new security mechanisms introduced more overhead than expected. Thus, SeDyA illustrates a trade-off between high security and the severe bandwidth usage of forwarding all beacon messages, and the lower level of security but high bandwidth efficiency of existing secure aggregation schemes. However, it should be noted that SeDyA achieves the dynamic aggregation goal to reduce the bandwidth requirements in the dissemination phase. Future work will be able to determine the potential of this approach by performing large scale simulations and further study the trade-off between bandwidth and disseminated information, while maintaining a high level of security. The thesis has considered the following research questions, which will be discussed individually:

1. How can we provide data integrity and consistency for in-network aggregation in VANETs?
2. Which guarantees and how much confidence can we gain by using cryptography for data integrity and consistency in VANET aggregation?
3. How can we compare different secure aggregation schemes in terms of accuracy, privacy and security guarantees?
4. How can we ensure that aggregation schemes can scale to a sufficiently large area without loosing too much accuracy?
5. Can we obtain a better trade-off between security and privacy than the current state of the art in VANET aggregation?

1. As already discussed to some extent by related work, both integrity and consistency can be reached in several ways, using trust and consistency checks, cryptographic primitives and interactive verification. The most common approach for consistency is to rely on trust and checks, because this typically does not incur high overhead. However, as has been shown by some secure aggregation schemes, cryptographic primitives can be used to support this approach at the cost of additional overhead. This master thesis has combined consistency checks with improved cryptographic primitives to build SeDyA, which provides a strong resilience against attacks by relying on both these mechanisms. In addition, SeDyA also has a component of interactive verification, because a message is first aggregated and then disseminated through the aggregation area for signing. Finally, the interested vehicles that receive the messages are protected against remote impersonation attacks, due to the fact that the pseudonyms of each signer of a message are known. Due to this knowledge, the message can be used for further consistency checks, to allow vehicles to resolve conflicting information.

2. Unfortunately, the guarantees offered by the cryptographic mechanisms introduced by related work are not as strong as previously assumed. In particular, this thesis has shown that the approach used by

SAS and by AM-FM sketches is vulnurable to some attacks, when converted into a more realistic VANET scenario. SeDyA has been developed with these attacks in mind, aiming to combine consistency checks with cryptographic integrity to improve the confidence that vehicles can have in a message. Specifically, it provides a number of signers, which can be used to make the computation of the confidence in this message explicit. In addition, SeDyA provides more protection against attacks, compared to the alternatives, particularly when those attacks are frequent. However, SeDyA is still vulnurable to a larger scale collusion of distinct attackers, which declared out of scope by SeDyA as well as other secure aggregation schemes from literature.

3. The comparison of different guarantees concerning accuracy, privacy and security, as well as feasability, has proven more difficult than intially expected. To the authors knowledge, there are no well-established metrics to determine these results; in addition, the mechanisms typically rely on different assumptions that make direct comparison challenging at best. While analyzing these protocols by modifying simulations is possible, there are typically no implementations available for the related work, making it an extremely time-consuming and error-prone process to perform this analysis. Therefore, the simulations performed for the new scheme, SeDyA, have been compared against a limited version that is closely related to two schemes from related work. The evaluation metrics are then designed in a manner similar to related work, using some new but well-documented metrics and some existing metrics from other works.

4. The scalability of aggregation is one of the main reasons to construct SeDyA, which builds on elements of related work to create a more secure and dynamic aggregation protocol. To the knowledge of the author, this is the first general aggregation scheme that provides both security and a dynamically defined area. Defining the area in a dynamic fashion is essential to providing a good and scalable aggregation scheme, because it reduces the amount of aggregates that need to be disseminated as the area size grows. Related work that considers security relies on a fixed segment size, which may needlessly bound aggregation or cause inaccurate aggregates to be generated when, for example, the bound of a traffic jam falls in the middle of such a fixed segment. While the evaluation has shown that SeDyA's intended design works, providing this secure scheme incurs significant bandwidth overhead. However, future work should be able to adapt SeDyA to support more flexible and less bandwidth-consuming mechanisms. This may be possible by stripping away the security of SeDyAs aggregation phase entirely, or by using efficient message dissemination schemes in the finalization phase. However, this approach may incur a cost in terms of security, and should be carefully analyzed before being applied in practice. Nevertheless, this shows that SeDyA provides a different choice for the trade-off between security, bandwidth usage and accuracy.

5. As per the results from Chapter 6, it seems that SeDyA does not provide an improvement of three elements in this trade-off between security, accuracy and bandwidth usage. Instead, the security is improved, at the cost of a significant increase in bandwidth consumption. This clearly shows that it is possible to provide stronger security in the presence of independent attackers. Note that, as discussed in Section 6.5, there are still some approaches that have not been explored. These approaches typically involve concessions in security to achieve better bandwidth efficiency, allowing further study of this trade-off.

To summarize, SeDyA provides a valuable contribution to the study of the trade-off between security, bandwith efficiency and accuracy. Future work should be able to further study this trade-off by introducing additional mechanisms to SeDyA's finalization phase, and by removing the security mechanism in the aggregation phase. Furthermore, future work should focus on other available combinations of consistency, cryptography and interactive mechanisms to improve the security and bandwidth consumption of secure aggregation schemes. While some of these mechanisms have been designed in previous work already, none of these schemes use cryptography as a means to combine the consistency checks of many vehicles. Finally, there is room to develop a more rigorous and formal analysis of the impact an attacker can have when consistency checks are a significant component in security. This also applies more generally to the use of consistency checks in VANETs, which are sometimes employed to argue against strong security mechanisms.

Bibliography

[1] IEEE trial-use standard for wireless access in vehicular environments - security services for applications and management messages, 2006. IEEE Std 1609.2-2006.

[2] A. Barenghi, G. Bertoni, L. Breveglieri, and G. Pelosi. A fpga coprocessor for the cryptographic tate pairing over fp. In *Information Technology: New Generations, 2008. ITNG 2008. Fifth International Conference on*, pages 112 –119, april 2008.

[3] R. Barr. *An efficient, unifying approach to simulation using virtual machines.* PhD thesis, Cornell University, May 2004.

[4] J.-L. Beuchat, N. Brisebarre, J. Detrey, and E. Okamoto. Arithmetic operators for pairing-based cryptography. 4727:239–255, 2007.

[5] N. Bissmeyer, C. Stresing, and K. Bayarou. Intrusion detection in vanets through verification of vehicle movement data. In *Vehicular Networking Conference (VNC), 2010 IEEE*, pages 166 –173, dec. 2010.

[6] A. Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In *Public Key Cryptography*, pages 31–46, 2003.

[7] D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. *SIAM Journal on Computing*, 32(3):586, 2003.

[8] D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *Advances in Cryptology - EUROCRYPT 2003*, volume 2656, page 416, 2003.

[9] CAMP Vehicle Safety Communications Consortium. Vehicle safety communications project (final report), May 2005.

[10] S. Chang, Y. Qi, H. Zhu, J. Zhao, and X. Shen. Footprint: Detecting sybil attacks in urban vehicular networks. *Parallel and Distributed Systems, IEEE Transactions on*, PP(99):1, 2012.

[11] S. Dashtinezhad, T. Nadeem, B. Dorohonceanu, C. Borcea, P. Kang, and L. Iftode. 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*, volume 5, pages 2946 – 2950 Vol.5, may 2004.

[12] S. Dietzel, B. Bako, E. Schoch, and F. Kargl. A fuzzy logic based approach for structure-free aggregation in vehicular ad-hoc networks. In *Proceedings of the sixth ACM international workshop on VehiculAr InterNETworking - VANET '09*, page 79, 2009.

[13] S. Dietzel, E. Schoch, B. Konings, M. Weber, and F. Kargl. Resilient secure aggregation for vehicular networks. *IEEE Network*, 24(1):26, 2010.

[14] D. Dolev and A. C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, IT-29(2):198–208, March 1983.

[15] J. R. Douceur. The sybil attack. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '01, pages 251–260, London, UK, 2002. Springer-Verlag.

[16] ETSI. Intelligent transport systems (ITS); vehicular communications; basic set of applications; definitions. `http://www.etsi.org/deliver/etsi_tr/102600_102699/102638/01.01.01_60/tr_102638v010101p.pdf`, June 2006. ETSI TR 102 638, version 1.1.1.

[17] Y.-C. Fan and A. L. Chen. Efficient and robust sensor data aggregation using linear counting sketches. In *2008 IEEE International Symposium on Parallel and Distributed Processing*, page 1, 2008.

[18] 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, pages 135–138, New York, NY, USA, 2012. ACM.

[19] P. Flajolet and G. N. Martin. Probabilistic counting. In *FOCS'83*, pages 76–82, 1983.

[20] M. Garofalakis, J. M. Hellerstein, and P. Maniatis. Proof sketches: Verifiable in-network aggregation. In *2007 IEEE 23rd International Conference on Data Engineering*, page 996, 2007.

[21] C. Gentry and Z. Ramzan. Identity-based aggregate signatures. In *Public Key Cryptography - PKC 2006, 9th International Conference on Theory and Practice of Public-Key Cryptography*, pages 257–273, 2006.

[22] M. Gerlach. Assessing and improving privacy in vanets. *ESCAR Embedded Security in Cars*, 2006.

[23] M. Gerlach. Assessing and improving privacy in VANETs. In *4th Workshop on Embedded Security in Cars*, 2006.

[24] Q. Han, S. Du, D. Ren, and H. Zhu. Sas: A secure data aggregation scheme in vehicular sensing networks. In *2010 IEEE International Conference on Communications*, page 1, 2010.

[25] H.-C. Hsiao, A. Studer, R. Dubey, E. Shi, and A. Perrig. Efficient and secure threshold-based event validation for vanets. In *WISEC'11*, pages 163–174, 2011.

[26] A. Joux and K. Nguyen. Separating decision diffie-hellman from diffie-hellman in cryptographic groups. Cryptology ePrint Archive, Report 2001/003, 2001. `http://eprint.iacr.org/`.

[27] Z. Li and G. Gong. Data aggregation integrity based on homomorphic primitives in sensor networks. In *ADHOC-NOW'10*, pages 149–162, 2010.

[28] X. Lin, X. Sun, P.-H. Ho, and X. Shen. Gsis: A secure and privacy-preserving protocol for vehicular communications. *Vehicular Technology, IEEE Transactions on*, 56(6):3442 –3456, nov. 2007.

[29] 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*, page 1, 2007.

[30] J. Lutz and A. Hasan. High performance fpga based elliptic curve cryptographic co-processor. In *International Conference on Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004.*, page 486, 2004.

[31] A. Menezes, T. Okamoto, and S. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *Information Theory, IEEE Transactions on*, 39(5):1639 –1646, sep 1993.

[32] A. Metwally, D. Agrawal, and A. E. Abbadi. Why go logarithmic if we can go linear? In *Proceedings of the 11th international conference on Extending database technology Advances in database technology - EDBT '08*, pages 618–629, 2008.

[33] 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*, pages 13 – 26, 2004.

[34] S. Nath, P. B. Gibbons, S. Seshan, and Z. R. Anderson. Synopsis diffusion for robust aggregation in sensor networks. In *Proceedings of the 2nd international conference on Embedded networked sensor systems - SenSys '04*, page 250, 2004.

[35] J. Petit and Z. Mammeri. Analysis of authentication overhead in vehicular networks. In *Wireless and Mobile Networking Conference (WMNC), 2010 Third Joint IFIP*, pages 1 –6, oct. 2010.

[36] 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, pages 67–75. ACM, 2006.

[37] M. Raya and J.-P. Hubaux. Securing vehicular ad hoc networks. *Journal of Computer Security*, 15(1):39–68, 2007.

[38] J. Rezgui and S. Cherkaoui. Detecting faulty and malicious vehicles using rule-based communications data mining. In *Local Computer Networks (LCN), 2011 IEEE 36th Conference on*, pages 827 –834, oct. 2011.

[39] 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*, page 285, 2009.

[40] E. Schoch, F. Kargl, and M. Weber. Communication patterns in vanets. *IEEE Communications Magazine*, 46(11):119, 2008.

[41] R. Schwartz, M. van Eenennaam, G. Karagiannis, G. Heijenk, W. Wolterink, and H. Scholten. Using v2v communication to create over-the-horizon awareness in multiple-lane highway scenarios. In *Intelligent Vehicles Symposium (IV), 2010 IEEE*, pages 998 –1005, june 2010.

[42] SeVeCom. SeVeCom Deliverable 2.1-App.A, version 1.2: Baseline security application, April 2009.

[43] A. Shamir. How to share a secret. *Commun. ACM*, 22:612–613, November 1979.

[44] V. Shoup. Practical threshold signatures. In *Proceedings of the 19th international conference on Theory and application of cryptographic techniques*, EUROCRYPT'00, pages 207–220, Berlin, Heidelberg, 2000. Springer-Verlag.

[45] A. Studer, F. Bai, B. Bellur, and A. Perrig. Flexible, extensible, and efficient vanet authentication. In *Proceedings of the 6th Embedded Security in Cars Workshop (ESCAR 08)*, November 2008.

[46] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu. The broadcast storm problem in a mobile ad hoc network. *Wirel. Netw.*, 8:153–167, March 2002.

[47] Y.-c. Tseng, S.-Y. Ni, Y.-s. Chen, and J.-P. Sheu. The Broadcast Storm Problem in a Mobile Ad Hoc Network. *Wireless Networks*, 8(2-3):153–167, 2002.

[48] University Ulm. Jist/swans extensions. http://vanet.info/node/12.html. Retrieved 9 July 2012.

[49] M. van Eenennaam and G. Heijenk. In *Proceedings of the Fourth International Workshop on Vehicle-to-Vehicle Communications, V2VCOM 2008*, pages 19–25, 2008.

[50] L. Wischoff, A. Ebner, H. Rohling, M. Lott, and R. Halfmann. Sotis - a self-organizing traffic information system. In *Vehicular Technology Conference, 2003. VTC 2003-Spring. The 57th IEEE Semiannual*, volume 4, pages 2442 – 2446 vol.4, april 2003.