Misbehavior Detection in Cooperative Intelligent Transport Systems

Dissertation zur Erlangung des Doktorgrades Dr. rer. nat. der Fakultät für Ingenieurwissenschaften, Informatik und Psychologie der Universität Ulm

Rens Wouter van der Heijden

aus 's-Hertogenbosch, Niederlande

2018

Institut für Verteilte Systeme
Universität Ulm, Deutschland

Icon & Image Attributions:

Car Icon by Iconnice from www.flaticon.com is licensed under Creative Commons BY 3.0
Antenna Icon by TagTeam Studio from the Noun Project
Antenna RSU Icon by Ruben Vh from the Noun Project

Amtierender Dekan: Prof. Maurits Ortmanns
Gutachter: Prof. Frank Kargl, Institute of Distributed Systems, Ulm University, Germany
Gutachter: Prof. Panagiotis Papadimitratos, Networked Systems Security Group, KTH Royal Institute of Technology, Sweden
Tag der Promotion: November 9th, 2018

Acknowledgments

Acknowledgements may be a terrible cliché, but no thesis is really complete without them. A thesis takes time, hard work, confidence, and persistence, among many other things. Although I consider it as a major achievement in my career, the truth is that this thesis would not have been possible without the support of those around me. Thus, I would like to thank the countless people that helped me, in ways big and small. This includes my family, my colleagues, my students, and of course the wonderful Natalie Dykes. I would also like to thank my fantastic co-authors for their help in writing and collaborating on a broad variety of papers. The anonymous reviewers of those papers of course also deserve some credit for their helpful feedback that improved the quality of the articles on which this thesis is based. I would specifically like to thank Dr. Stefan Dietzel for his supervision of my Master thesis, which brought me into contact with scientific work. Furthermore, I thank Prof. Dr.-rer-nat Frank Kargl for his supervising role and the large amount of academic freedom I was given, as well as his feedback on this thesis. Finally, I thank Benjamin Erb, Dominik Lang, and Matthias Matousek for taking the time to proofread parts of this thesis.

During my time in Ulm, I was lucky enough to participate actively in the institute's teaching activities, and I would like to thank all the students I taught and supervised for the interesting times. The following lists the students whose work was also in this field, and whose active discussions certainly contributed to the ideas presented in this thesis:

- Franziska Schönig (MSc, UUlm)
- David Köhler (MSc, UUlm)
- Florian Diemer (BSc, UUlm) [152]
- Johannes Diebold (BSc, UUlm)
- Christopher Keazor (Dipl, UUlm)
- Thai Chung (BSc, UUlm)
- Ala'a Al-Momani (MSc, UUlm & Princess Sumaya University for Technology, Jordan)
- Julian Schregle (BSc, UUlm)

The following students were part of Maat's development in the form of student assistant jobs (which were provided through the funding agencies listed below):

- Amir Hosh
- Yogita Suryawanshi
- Leo Hnatek
- Sergej Schmidt
- Johannes Diebold

I also had the pleasure of supervising a number of other students on various topics not directly related to this thesis:

- Mahmoud Nabegh (BSc, GUC)
- Witali Hepting (MSc, UUlm + Daimler TSS)
- Manuel Gassner (BSc, UUlm + Daimler TSS)
- Dominik Meißner (BSc, UUlm)
- Michael Ortmann (MSc, UUlm)
- Lukas Müller (BSc, UUlm)
- Christoph Rothermel (BSc, UUlm)
- Martin Deubzer (BSc, UUlm)

- Florian Unterstein (MSc, UUlm)
- Stefan Rotter (BSc, UUlm)
- Christian Forst (BSc, UUlm + Secunet)
- Felix Engelmann (BSc, UUlm)

Generous funding was provided by a number of sources over the years:

- Volkswagen AG within the CONVERGE project[1]
- The European FP7 Project CRISALIS[2]
- The Baden-Württemberg Stiftung for funding AutoDetect[3]
- Teaching positions at Ulm University

Part of this work was performed on the computational resource bwUniCluster funded by the Ministry of Science, Research and the Arts Baden-Württemberg and the Universities of the State of Baden-Württemberg, Germany, within the framework program bwHPC.

[1] http://www.converge-online.de/
[2] http://www.crisalis-project.eu/
[3] https://www.uni-ulm.de/in/vs/res/proj/autodetect/

Overview of Publications

The following articles are peer-reviewed publications with results included in this thesis (ordered from oldest to newest):

- S Dietzel, RW van der Heijden, H Decke, and F Kargl. "A Flexible, Subjective Logic-based Framework for Misbehavior Detection in V2V Networks". In: *15th International Symposium on a World of Wireless, Mobile and Multimedia Networks*. WoWMoM. IEEE, June 2014, pages 1–6. DOI: 10.1109/WoWMoM.2014.6918989.

- RW van der Heijden, A Al-Momani, F Kargl, and OMF Abu-Sharkh. "Enhanced Position Verification for VANETs using Subjective Logic". In: *Proceedings of the 84th Vehicular Technology Conference*. VTC-Fall. IEEE, September 2016. URL: http://arxiv.org/abs/1703.10399.

- A Al-Momani, RW van der Heijden, F Kargl, and C Waldschmidt. "Exploiting Propagation Effects for Authentication and Misbehavior Detection in VANETs". In: *Vehicular Networking Conference*. VNC. IEEE, December 2016, pages 1–4. DOI: 10.1109/VNC.2016.7835973.

- RW van der Heijden, T Lukaseder, and F Kargl. "Analyzing Attacks on Cooperative Adaptive Cruise Control (CACC)". in: *Vehicular Networking Conference*. VNC. Best paper award. IEEE, November 2017, pages 45–52. DOI: 10.1109/VNC.2017.8275598.

- RW van der Heijden, F Engelmann, D Mödinger, F Schönig, and F Kargl. "Blackchain: Scalability for Resource-Constrained Accountable Vehicle-to-X Communication". In: *Proceedings of the 1st Workshop on Scalable and Resilient Infrastructures for Distributed Ledgers*. SERIAL. ACM, 2017. DOI: 10.1145/3152824.3152828.

- RW van der Heijden, H Kopp, and F Kargl. "Multi-Source Fusion Operations in Subjective Logic". In: *International Conference on Information Fusion*. FUSION. IEEE, July 2018. URL: https://arxiv.org/abs/1805.01388.

- RW van der Heijden, T Lukaseder, and F Kargl. "VeReMi: A Dataset for Comparable Evaluation of Misbehavior Detection in VANETs". In: *Proceedings of the 14th EAI International Conference on Security and Privacy in Communication Networks*. SecureComm. Springer, August 2018. URL: https://arxiv.org/abs/1804.06701.

- RW van der Heijden, S Dietzel, T Leinmüller, and F Kargl. "Survey on Misbehavior Detection in Cooperative Intelligent Transportation Systems". In: *IEEE Communication Surveys & Tutorials* (2018). DOI: 10.1109/COMST.2018.2873088.

The following articles also contribute to this thesis, but were not peer-reviewed:

- RW van der Heijden, S Dietzel, and F Kargl. "Misbehavior Detection in Vehicular Ad-hoc Networks". In: *Proceedings of the 1st GI/ITG KuVS Fachgespräch Inter-Vehicle Communication*. FG-IVC. 2013.

- RW van der Heijden and F Kargl. "Open Issues in Data-Centric Misbehavior Detection for VANETs". In: *Proceedings of the 2nd GI/ITG KuVS Fachgespräch Inter-Vehicle Communication*. FG-IVC. University of Luxembourg, Interdisciplinary Centre for Security, Reliability and Trust (SnT), 2014, pages 24–26.

- M Dacier, F Kargl, H König, and A Valdes. "Network Attack Detection and Defense: Securing Industrial Control Systems for Critical Infrastructures (Dagstuhl Seminar 14292)". In: *Dagstuhl Reports* 4.7 (2014). Edited by M Dacier, F Kargl, H König, A Valdes, and RW van der Heijden, pages 62–79. DOI: 10.4230/DagRep.4.7.62.

- F Kargl, RW van der Heijden, H König, A Valdes, and M Dacier. "Dagstuhl Manifesto (Network Attack Detection and Defense: Securing Industrial Control Systems for Critical Infrastructures)". In: *Informatik Spektrum* 47.6 (2014), pages 605–607.

- F Kargl, RW van der Heijden, H König, A Valdes, and M Dacier. "Insights on the Security and Dependability of Industrial Control Systems". In: *IEEE Security & Privacy* 12.6 (November 2014), pages 75–78. DOI: 10.1109/MSP.2014.120.

- RW van der Heijden and F Kargl. "Evaluating Misbehavior Detection for Vehicular Networks". In: *Proceedings of the 5th GI/ITG KuVS Fachgespräch Inter-Vehicle Communication*. FG-IVC. Friedrich-Alexander-Universität Erlangen-Nürnberg, Department of Computer Science, 2017, pages 5–8.

The following list includes other projects performed in parallel to this thesis:

- RW van der Heijden, S Dietzel, and F Kargl. "SeDyA: Secure Dynamic Aggregation in VANETs". In: *Proceedings of the sixth ACM conference on Security and privacy in wireless and mobile networks*. WiSec. ACM, April 2013, pages 131–142. DOI: 10.1145/2462096.2462119.

- F Engelmann, T Lukaseder, B Erb, RW van der Heijden, and F Kargl. "Dynamic Packet-Filtering in High-Speed Networks using NetFPGAs". In: *Third International Conference on Future Generation Communication Technologies*. FGCT. Best paper award. IEEE, August 2014, pages 55–59. DOI: 10.1109/FGCT.2014.6933224.

- S Dietzel, J Gürtler, RW van der Heijden, and F Kargl. "Redundancy-based Statistical Analysis for Insider Attack Detection in VANET Aggregation Schemes". In: *Vehicular Networking Conference*. VNC. IEEE, December 2014, pages 135–142. DOI: 10.1109/VNC.2014.7013332.

- M Fazouane, H Kopp, RW van der Heijden, D Le Métayer, and F Kargl. "Formal Verification of Privacy Properties in Electric Vehicle Charging". In: *Engineering Secure Software and Systems*. Edited by F Piessens, J Caballero, and N Bielova. Cham: Springer International Publishing, 2015, pages 17–33. DOI: 10.1007/978-3-319-15618-7_2.

- S Kleber, RW van der Heijden, H Kopp, and F Kargl. "Terrorist Fraud Resistance of Distance Bounding Protocols Employing Physical Unclonable Functions". In: *2015 International Conference and Workshops on Networked Systems*. NetSys. IEEE, March 2015, pages 1–8. DOI: 10.1109/NetSys.2015.7089068.

- S Dietzel, RW van der Heijden, J Petit, and F Kargl. "Context-Adaptive Detection of Insider Attacks in VANET Information Dissemination Schemes". In: *Vehicular Networking Conference*. VNC. IEEE, December 2015, pages 287–294. DOI: 10.1109/VNC.2015.7385590.

- D Meißner, B Erb, RW van der Heijden, K Lange, and F Kargl. "Mobile Triage Management in Disaster Area Networks Using Decentralized Replication". In: *Proceedings of the Eleventh ACM Workshop on Challenged Networks*. CHANTS. ACM, 2016, pages 7–12. DOI: 10.1145/2979683.2979689.

- T Lukaseder, L Bradatsch, B Erb, RW van der Heijden, and F Kargl. "A Comparison of TCP Congestion Control Algorithms in 10G Networks". In: *41st Conference on Local Computer Networks*. LCN. IEEE, November 2016, pages 706–714. DOI: 10.1109/LCN.2016.121.

- T Lukaseder, A Hunt, C Stehle, D Wagner, RW van der Heijden, and F Kargl. "An Extensible Host-Agnostic Framework for SDN-Assisted DDoS-Mitigation". In: *42nd Conference on Local Computer Networks*. LCN. IEEE, October 2017, pages 619–622. DOI: 10.1109/LCN.2017.103.

- M Matousek, M Yassin, A Al-Momani, RW van der Heijden, and F Kargl. "Robust Detection of Anomalous Driving Behavior". In: *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*. June 2018, pages 1–5. DOI: 10.1109/VTCSpring.2018.8417777.

Abstract

Automobile accidents are one of the major causes of death in the Western world. In previous decades, automobile manufacturers and researchers have investigated a broad spectrum of solutions to this challenge. Within this solution space, communication between vehicles has long been a promising direction that enables highly advanced driver assistance systems. Current generation assistance systems operate through the use of automotive sensors, which have limited range; to provide the vehicle with a more complete picture of its surroundings, various standards have been proposed to enable information exchange between vehicles. Recent developments in this field, which integrate more components into this communication architecture, give rise to cooperative intelligent transport systems (C-ITS). Most C-ITS applications, in particular including safety applications, make decisions based on both information received from local sensors and messages received from others.

One aspect of C-ITS that is essential for successful deployment is its security against invalid behavior and malicious attacks. Without such protection, the validity of the information received from other vehicles cannot be guaranteed, and thus the reliability of all C-ITS applications is affected. Research has invested significant effort in the development of basic security services, such as pseudonymization and sender authentication. One area that has received limited attention in standardization is that of misbehavior by authentic entities in the network. For example, a malicious vehicle may transmit false messages, triggering an emergency response and causing a collision between other vehicles. This cannot be prevented through standard security services, such as cryptographic signatures, because a malicious vehicle is an authentic sender. In general the detection of such invalid application data is termed misbehavior detection. Because different attacks are typically detected through different misbehavior detection mechanisms, the combination of these outputs (i.e., fusion) for decision making is an essential component. This thesis addresses this topic by designing Maat, a generic misbehavior detection framework that ensures the validity of received data.

The contributions of this thesis include (a) a detailed survey of existing misbehavior detection mechanisms, (b) Maat, a proposal for a generic fusion framework for misbehavior detection in C-ITS, (c) multi-source fusion operations for subjective logic, which forms the mathematical foundation of our framework, (d) several novel detection mechanisms, (e) a detailed review of evaluation methodologies and proposals for novel metrics, (f) a new, public dataset that serves as a baseline for comparison of misbehavior detection mechanisms, (g) a detailed evaluation of the proposed mechanisms and fusion operations, and (h) an outlook discussing how these results can be applied to other cyber-physical systems.

The survey in this thesis provides an overview and classification of existing misbehavior detection mechanisms along various axes, including the scope of detection, type of data used and susceptibility to attacks. Not only does this provide a solid foundation for the requirements on Maat, it also supports the development of attacks and misbehavior detection mechanisms in the wider field.

Within this thesis, we build a framework, called Maat, to fuse misbehavior detection results through subjective logic. Subjective logic is a mathematical framework that enables the expression of uncertainty on data through objects called subjective opinions. Maat applies this logic to build a flexible data management and fusion system, which determines the trustworthiness of data whenever it is accessed by applications. To support this data management, Maat uses a directed graph to store the data and the associated detection results. By recording both the data and the associated detection results separately, a wide range of potential new detectors can be explored. In addition, it enables the verifiable exchange of detection results for revocation.

Subjective logic provides a variety of fusion operators to fuse subjective opinions. However, for some of these operators, fusion of multiple opinions (multi-source fusion) is not well-defined due to non-commutativity. In order to implement Maat, these operators were generalized to the multi-source fusion

setting: we provide this generalization for weighted belief fusion (WBF) and consensus & compromise fusion (CCF). We also discuss how transitive trust relations can be applied within our framework.

Maat contains a set of new detection mechanisms that exploit properties of subjective logic to more accurately model the detection results. We use these mechanisms to show that fusion can increase detection performance compared to individual detection mechanisms.

As part of our survey of related work, we found that there are significant methodological differences and evaluation criteria. In this thesis, we provide an overview of those differences, and propose a new evaluation methodology that goes significantly beyond the rigor exhibited by existing work. This methodology includes a set of application-centric metrics for cooperative adaptive cruise control, one of the primary C-ITS applications, as well as metrics to assess overall detection performance in a widely deployed system.

One issue we encountered in reproducing the work of others is the fact that there are no publicly available benchmarks against which misbehavior detection mechanisms can be tested. In this thesis, we present a public dataset that can serve as a baseline for such benchmarks.

Based on this new methodology and the presented dataset, we provide a detailed evaluation of Maat's features. This includes a study of detection performance by different detection mechanisms, a comparison of fusion operations, and the analysis of weighing between detectors. We also revisit the idea of exponential weighted averaging (EWA) of detection output to protect against accidental faults. Our results show that Maat can provide an overall improvement in detection performance, while the EWA reduces performance even when attacks are persistently executed. We attribute this failure of EWA to the types of attacks executed in our experiments, whose detection depends on the spatial relationship between attacker and observer. This evidence suggests that EWA is not suitable in these specific scenarios.

In summary, this thesis studies the topic of misbehavior detection in cooperative intelligent transport systems. Misbehavior detection exploits knowledge of physical processes to determine the trustworthiness of data and entities in a cyber-physical system. Through our developed fusion framework for misbehavior detection mechanisms, the safety and security of such systems can be improved significantly. Future work in this field could includes the integration of misbehavior detection with sensor fusion processes to validate sensor data and protect against attacks on such systems, as well as extensions that enable reliable reporting and sharing of parts of Maat's world model.

Samenvatting

Autoongevallen zijn een van de belangrijkste doodsoorzaken in de westerse wereld. In de afgelopen decennia hebben autofabrikanten en onderzoekers een breed spectrum aan oplossingen voor deze uitdaging onderzocht. Binnen deze oplossingsruimte is de communicatie tussen voertuigen al lange tijd een veelbelovende richting die geavanceerde bestuurdersassistentiesystemen mogelijk maakt. Om het voertuig een vollediger beeld van zijn omgeving te geven, zijn verschillende normen voorgesteld om de uitwisseling van informatie tussen voertuigen mogelijk te maken. Recente ontwikkelingen op dit gebied, waarbij meer componenten in deze communicatiearchitectuur worden geïntegreerd, leiden tot coöperatieve intelligente vervoerssystemen (C-ITS). De meeste C-ITS-toepassingen, met name veiligheidstoepassingen, nemen beslissingen op basis van zowel informatie van lokale sensoren als berichten van anderen.

Een aspect van C-ITS dat essentieel is voor een succesvolle implementatie is de beveiliging ervan tegen ongeldig gedrag en kwaadwillige aanvallen. Zonder een dergelijke bescherming kan de geldigheid van de van andere voertuigen ontvangenen informatie niet worden gegarandeerd, waardoor de betrouwbaarheid van alle C-ITS-toepassingen wordt aangetast. In het onderzoek van de laatste jaren is veel geïnvesteerd in de ontwikkeling van basisbeveiligingsdiensten, zoals pseudonimisering en authenticatie van afzenders. Een gebied dat bij standaardisatie beperkte aandacht heeft gekregen, is dat van wangedrag door autentische entiteiten in het netwerk. Een kwaadaardig voertuig kan bijvoorbeeld foute berichten verzenden, waardoor een noodmaatregel in een auto wordt geactiveerd en een botsing tussen andere voertuigen wordt veroorzaakt. Dit kan niet worden voorkomen door standaardveiligheidsdiensten, zoals cryptografische handtekeningen, omdat een kwaadaardig voertuig een autentische afzender is. In het algemeen wordt de detectie van dergelijke ongeldige applicatiegegevens aangeduid als misbehavior detection. Omdat verschillende aanvallen doorgaans worden gedetecteerd door verschillende misbehavior detection mechanismen, is de combinatie van deze uitgangen (d.w.z. fusie) voor de besluitvorming een essentieel onderdeel. Dit proeschrift pakt dit onderwerp aan met de ontwikkeling van Maat, een generisch framework voor het opsporen van wangedrag dat de validiteit van ontvangen gegevens waarborgt.

De bijdragen van deze dissertatie omvatten (a) een gedetailleerd overzicht van bestaande misbehavior detection mechanismen, (b) Maat, een voorstel voor een generiek fusie-framework voor misbehavior detection in C-ITS, (c) multi-source fusieoperaties voor subjective logic, die de wiskundige basis van ons framework vormt, (d) verschillende nieuwe detectiemechanismen, e) een gedetailleerd overzicht van de evaluatiemethodologieën en voorstellen voor nieuwe meetmethoden, f) een nieuwe, openbare dataset die als basis dient voor de vergelijking van misbehavior detection mechanismen, g) een gedetailleerde evaluatie van de voorgestelde mechanismen en fusieoperaties, en h) een vooruitzicht waarin wordt besproken hoe deze resultaten op andere cyberfysische systemen kunnen worden toegepast.

De literatuur-analyse in dit proefschrift biedt een overzicht en classificatie van bestaande detectiemechanismen voor wangedrag langs verschillende assen, waaronder de reikwijdte van de detectie, het type gegevens dat wordt gebruikt en de gevoeligheid voor aanvallen. Dit biedt niet alleen een solide basis voor de eisen aan Maat, het ondersteunt ook de ontwikkeling van aanvallen en wangedrag detectiemechanismen in het bredere veld.

Dit proefschrift behandelt de ontwikkeling van een framework genaamd Maat dat de resultaten van wangedragdetectie fuseert met subjective logic. In subjective logic wordt de onzekerheid in data meegenomen door gebruik te maken van zogenaamde subjective opinions. Maat gebruikt deze logic om een flexibel datamanagement en fusiesysteem te bouwen dat de betrouwbaarheid van data weergeeft als applicaties er gebruik van maken. Ter ondersteuning van het fusiesysteem geeft Maat deze gegevens en de bijbehorende resultaten van detectoren in een gerichte graaf weer. Het vastleggen van zowel de data als de bijbehorende detectieresultaten maakt het mogelijk om een breed scala aan potentieel nieuwe detectoren te onderzoeken. Bovendien kan men dan detectieresultaten controleerbaar uitwisselen, bijvoorbeeld ten

behoeve van het intrekken van certificaten.

Subjective logic biedt een breed scala aan fusieoperatoren om subjective opinions te fuseren. Maar voor sommige van deze operatoren is de fusie van meerdere opinions (multi-source fusion) niet eenduidig gedefinieerd omdat ze niet commutatief zijn. Om Maat te kunnen implementeren werden deze operatoren veralgemeniseerd naar een multi-source fusion setting: we vertaalden de generalisatie voor de weighted belief fusion (WBF) de consensus & compromise fusion (CCF). We bespreken ook hoe transitieve vertrouwensrelaties in ons framework kunnen worden toegepast.

Maat bevat een nieuwe set detectiemechanismen die gebruik maken van de eigenschappen van subjective logic om de detectieresultaten beter te kunnen modelleren. We gebruiken deze mechanismen om aan te tonen dat fusie de detectieprestaties kan verbeteren in vergelijking met individuele detectiemechanismen.

Tijdens ons onderzoek hebben we vastgesteld dat er aanzienlijke methodologische verschillen en evaluatiecriteria bestaan. In dit proefschrift geven we daarvan een overzicht en stellen we een nieuwe evaluatiemethode voor om de analyse van aanvallen te verbeteren tegenover de nu gebruikelijke methodes. Deze methodologie omvat een reeks toepassingsgerichte metrieken voor cooperative adaptive cruise control, één van de primaire C-ITS-toepassingen, evenals metrieken om de algemene detectieprestaties bij grootschalige toepassing te kunnen beoordelen. Verder presenteren we bij dit proefschrift een openbare dataset die als basis kan dienen voor dit soort benchmarks.

Met behulp van deze methode en de dataset hebben we Maat tot in detail kunnen evalueren. Dat omvat een studie naar de prestaties van verschillende detectiemechanismen, een vergelijking van de fusieoperaties en een analyse van de weging van verschillende detectoren. We hebben ook het idee onderzocht om toevallige fouten in detectoren te minimaliseren door gebruik te maken van exponential weighted averaging (EWA). Uit onze resultaten blijkt dat, terwijl Maat kan zorgen voor een algemene verbetering van de detectieprestaties, EWA die vermindert, zelfs wanneer aanvallen voortdurend worden uitgevoerd. Wij denken dat het falen van EWA in onze experimenten toe te schrijven is aan het feit dat de detectie afhankelijk is aan de ruimtelijke relatie tussen aanvaller en waarnemer. Dit wijst erop dat EWA niet geschikt zijn voor deze specifieke scenario's.

Samenvattend behandelt dit proefschrift het onderwerp misbehavior detection in coöperatieve intelligente transportsystemen. Dat maakt gebruik van de kennis over fysieke processen om de betrouwbaarheid van gegevens en entiteiten in een cyberfysiek systeem te bepalen. Met het door ons ontwikkelde fusion framework voor misbehavior detection mechanismen kan de veiligheid van dergelijke systemen aanzienlijk worden verbeterd. Een toekomstige activiteit op dit gebied zou integratie van misbehavior detection met sensorfusieprocessen kunnen zijn. Dat zou het mogelijk maken om sensorgegevens te valideren om dergelijke systemen tegen aanvallen te beschermen. Ook zouden uitbreidingen die een betrouwbare rapportage en de uitwisseling van delen van het wereldmodel Maat mogelijk maken, zinvol zijn.

Zusammenfassung

Autounfälle sind eine der Haupttodesursachen in der westlichen Welt. In den vergangenen Jahrzehnten haben Automobilhersteller und Forschende ein breites Spektrum an Lösungen für diese Herausforderung untersucht. Innerhalb dieses Lösungsraumes ist die Kommunikation zwischen den Fahrzeugen seit langem eine vielversprechende Richtung, die hochmoderne Fahrerassistenzsysteme ermöglicht. Die Assistenzsysteme der heutigen Generation arbeiten mit Hilfe von Sensoren, deren Reichweite begrenzt ist; um dem Fahrzeug ein vollständigeres Bild seiner Umgebung zu vermitteln, wurden verschiedene Standards vorgeschlagen, die den Informationsaustausch zwischen den Fahrzeugen ermöglichen. Jüngste Entwicklungen in diesem Bereich, die mehr Komponenten in diese Kommunikationsarchitektur integrieren, führen zu kooperativen intelligenten Verkehrssystemen (C-ITS). Die meisten C-ITS-Anwendungen, insbesondere auch Safety-anwendungen, treffen Entscheidungen aufgrund von Informationen, die sowohl von lokalen Sensoren als auch von anderen Fahrzeugen empfangen werden.

Für den erfolgreichen Einsatz von C-ITS ist die Absicherung gegen ungültiges Verhalten der teilnehmenden Fahrzeuge, sowie gegen bösartige Angriffe, essentiell. Ohne einen solchen Schutz kann die Gültigkeit der von anderen Fahrzeugen erhaltenen Informationen nicht garantiert werden, so dass die Zuverlässigkeit aller C-ITS-Anwendungen beeinträchtigt wird. Die Forschung hat erhebliche Anstrengungen in die Entwicklung grundlegender Sicherheitsfunktionen wie Pseudonymisierung und Absenderauthentifizierung investiert. Ein Bereich, welcher bei der Standardisierung bisher wenig Beachtung gefunden hat, ist das Fehlverhalten von authentischen Entitäten im Netzwerk. Beispielsweise kann ein bösartiges Fahrzeug versuchen falsche Nachrichten zu übermitteln, die eine Notfallreaktion auslösen und eine Kollision zwischen anderen Fahrzeugen verursachen. Dies kann nicht durch Standard-Sicherheitsfunktionen wie Signaturen verhindert werden, da ein bösartiges Fahrzeug ein authentischer Absender ist. Im Allgemeinen wird die Erkennung solcher ungültiger Anwendungsdaten als Misbehavior Detection bezeichnet. Da verschiedene Angriffe typischerweise durch unterschiedliche Misbehavior Detection Mechanismen erkannt werden, ist die Kombination dieser Ausgänge (i.e. Fusion) für die Entscheidungsfindung ein wesentlicher Bestandteil. Diese Arbeit befasst sich mit diesem Thema durch die Entwicklung von Maat, einem generischen Framework zur Misbehavior Detection, welches die Gültigkeit der empfangenen Daten sicherstellt.

Die Beiträge dieser Arbeit umfassen (a) einen detaillierten Überblick über bestehende Misbehavior Detection Mechanismen, (b) Maat, einen Vorschlag für ein generisches Fusionsframework zur Misbehavior Detection in C-ITS, (c) Multi-Source-Fusionsoperationen für Subjective Logic, die das mathematische Fundament unseres Frameworks bilden, (d) mehrere neue Misbehavior Detection Mechanismen, (e) eine detaillierte Überprüfung der Bewertungsmethoden und Vorschläge für neue Metriken, (f) einen neuen, öffentlichen Datensatz, der als Grundlage für den Vergleich von Misbehavior Detection Mechanismen dient, (g) eine detaillierte Bewertung der vorgeschlagenen Mechanismen und Fusionsverfahren und (h) einen Ausblick, wie diese Ergebnisse auf andere cyberphysikalische Systeme angewendet werden können.

Die Literaturübersicht in dieser Arbeit bietet einen Überblick und eine Klassifizierung der bestehenden Misbehavior Detection Mechanismen entlang verschiedener Achsen, einschließlich der Lokalität der Erkennung, der Art der verwendeten Daten und der Anfälligkeit für Angriffe. Dies stellt nicht nur eine solide Grundlage für die Anforderungen an Maat dar, sondern unterstützt auch die Entwicklung von Angriffen und Misbehavior Detection Mechanismen im weiteren Umfeld.

Innerhalb dieser Arbeit entwickeln wir ein Framework, Maat, um Ergebnisse von Misbehavior Detection Mechanismen durch Subjective Logic zu fusionieren. Subjective Logic ist ein mathematisches Framework, welches den Ausdruck von Unsicherheit über Daten ermöglicht mittels so genannten Subjective Opinions. Maat verwendet diese Logik, um ein flexibles Datenmanagement- und Fusionssystem aufzubauen, das die Vertrauenswürdigkeit der Daten bei jedem Zugriff durch Anwendungen bes-

timmt. Zur Unterstützung dieses Datenmanagements verwendet Maat einen gerichteten Graphen zur Speicherung der Daten und der zugehörigen Erkennungsergebnisse. Durch die getrennte Erfassung der Daten und der dazugehörigen Detektionsergebnisse kann eine Vielzahl von potenziellen neuen Detektoren untersucht werden. Darüber hinaus ermöglicht es den nachweisbaren Austausch von Erkennungsergebnissen, zum Beispiel für die Revocation von Zertifikaten.

Subjective Logic bietet eine Vielzahl von Fusionsoperatoren, um Subjective Opinions zu fusionieren. Für einige dieser Operatoren ist die Fusion mehrerer Meinungen (Multi-Source-Fusion) jedoch aufgrund der Nicht-Kommutativität nicht eindeutig definiert. Um Maat zu implementieren, wurden diese Operatoren auf die Multi-Source-Fusionseinstellung verallgemeinert: Wir bieten neue Verallgemeinerungen für die Weighed Belief Fusion (WBF) und die Consensus & Compromise Fusion (CCF). Wir diskutieren auch, wie transitive Vertrauensbeziehungen in unserem Framework angewendet werden.

Maat enthält eine Reihe neuer Erkennungsmechanismen, die die Eigenschaften der Subjective Logic nutzen, um die Erkennungsergebnisse genauer zu modellieren. Wir nutzen diese Mechanismen um zu zeigen dass die Fusion die Erkennungsleistung im Vergleich zu einzelnen Erkennungsmechanismen steigern kann.

Im Rahmen unserer Literaturarbeit haben wir festgestellt, dass es erhebliche methodische Unterschiede und Bewertungskriterien gibt. In dieser Doktorarbeit geben wir einen Überblick über diese Unterschiede und schlagen eine neue Methodik vor, die hinsichtlich ihrer Stringenz deutlich über bestehender Arbeiten hinausgeht. Diese Methodik umfasst eine Reihe von anwendungszentrierten Metriken für die Cooperative Adaptive Cruise Control, eine der wichtigsten C-ITS-Anwendungen, sowie Metriken zur Bewertung der gesamten Erkennungsleistung in einem weit verbreiteten System.

Ein Problem, auf das wir bei der Reproduktion bestehender Arbeiten von anderen Forschern gestoßen sind, ist die Tatsache, dass es keine öffentlich zugänglichen Benchmarks gibt, gegen die Misbehavior Detection Mechanismen getestet werden können. In dieser Arbeit stellen wir einen öffentlichen Datensatz vor, der als Grundlage für solche Benchmarks dienen kann.

Basierend auf dieser neuen Methodik und dem vorgestellten Datensatz bieten wir eine detaillierte Auswertung der Features von Maat. Dies beinhaltet eine Untersuchung der Erkennungsleistung durch verschiedene Misbehavior Detection Mechanismen, einen Vergleich der verfügbaren Fusionsoperationen und die Analyse der Gewichtung zwischen den Detektoren. Auch die Idee von exponential weighted averaging (EWA) der Detektionsausgaben zum Schutz vor versehentlichen Fehlern wird aufgegriffen. Unsere Ergebnisse zeigen, dass Maat die Erkennungsleistung insgesamt verbessern kann, während die Erkennungsleistung von EWA sogar auch bei durchgehenden Angriffen reduziert wird. Wir führen den Misserfolg von EWA auf die Art der Angriffe in unseren Experimenten zurück, deren Erkennung von der räumlichen Beziehung zwischen Angreifer und Beobachter abhängt. Diese Erkenntnisse deuten darauf hin, dass EWA in diesen speziellen Szenarien nicht geeignet ist.

Zusammenfassend wird in dieser Arbeit das Thema Misbehavior Detection in kooperativen intelligenten Verkehrssystemen untersucht. Die Erkennung von Fehlverhalten nutzt das Wissen über physikalische Prozesse, um die Vertrauenswürdigkeit von Daten und Entitäten in einem cyberphysikalischen System zu ermitteln. Durch unser entwickeltes Fusionsframework für Misbehavior Detection Mechanismen kann die Sicherheit solcher Systeme erheblich verbessert werden. Zukünftige Arbeiten in diesem Bereich könnten die Integration von Misbehavior Detection mit Sensorfusionsprozessen zur Validierung von Sensordaten und zum Schutz vor Angriffen auf solche Systeme sowie, Erweiterungen umfassen, die ein zuverlässiges Reporting und die gemeinsame Nutzung von Teilen des Maat-Weltmodells ermöglichen.

Contents

Part I

Introduction

Chapter 1

Introduction

This chapter provides an overview of misbehavior detection and cooperative intelligent transport systems, as well as giving a broad overview of existing work in this area. Using this overview as a foundation, the research questions addressed in this thesis are then introduced, alongside a problem statement. Parts of this chapter have previously been published as part of a literature survey published by IEEE Communication Surveys & Tutorials [142, 170].

1.1 Motivation

Throughout the field of computer science, securing systems against malicious attackers has become a fundamental requirement for safe, secure, and dependable operation of applications. Today, professional attacks against systems, which are mounted by large criminal organizations or even governments, are becoming increasingly common [119, 135]. At the same time, computer systems are increasingly intertwined with the real world, making them more appealing targets. The term cyber-physical systems (CPSs) has been coined to encompass systems that are characterized by a large deployment of networked devices equipped with both sensors and actuators [120]. They are distinguished from traditional embedded systems, where individual nodes interact with the real world in strongly constrained environments. In contrast, CPSs are highly networked, deployed in large regions, and may contain nodes with heterogeneous computational power. A prominent example of a CPS is a cooperative intelligent transport system (C-ITS), which consists of vehicles, road-side units and back-end systems, and which is the focus of this thesis. The content transferred in CPSs and C-ITSs is highly predictable, relating directly to real-world phenomena [96, 120], a fact that enables novel techniques to detect attacks, collectively referred to as *misbehavior detection*. Attack detection in general is an essential second layer of security for networks, especially for widely deployed networked systems in potentially hostile environments, where attackers may have physical access to a subset of the system. Furthermore, the impact of such attacks is much greater, as they can easily be tailored to cause real-world harm or loss of life. Therefore, misbehavior detection in both CPSs and C-ITSs is essential for the secure and thus safe operation of these systems.

Cooperative Intelligent Transport Systems are networks designed to provide a variety of benefits [136, 143, 175]. These include improved road-safety, greener driving through improved traffic management, support for partially autonomous vehicles, and infotainment services such as traffic information services. The characterizing communication paradigm of all these applications is that vehicles observe real world conditions, such as positions of other vehicles or road conditions, which are then communicated over a ubiquitous network. This network is built up by equipping each vehicle with a wireless interface, creating a dynamic ad-hoc network that can be accessed without further overhead, which is commonly referred to as a *vehicular ad-hoc network (VANET)*. The VANET can also include infrastructure components, referred to as road-side unit (RSU), which are sparsely positioned along the road. The resulting network that includes sparse infrastructure is referred to as a *vehicular network*. Vehicles use the VANET to send and receive information, building a *world model* from received messages, which is used for the applications mentioned above. However, vehicles can also sense local information through a variety of sensors, especially with recent developments in partially autonomous driving. This information, communicated through vehicle-internal networks, is used for autonomous decision making by the vehicle, either in dedicated driving scenarios or with complete autonomy. These applications are often supported by

additional infrastructure in the back-end through the Internet. These extensions of a vehicular network, where cooperation between autonomous vehicles and back-end is central, are collectively referred to as C-ITSs. In addition to the benefits discussed above, C-ITSs are expected to support the deployment of autonomous driving technologies with cooperation between vehicles.

Both existing VANET applications and envisioned C-ITS applications present a host of new security challenges. One of these is the integrity and correctness of transmitted information that is exchanged between vehicles. A wide body of work is aiming to provide this protection, which can broadly be categorized into **proactive** and **reactive** mechanisms [22]. Proactive security aims to prevent potential attackers from system access, whereas reactive security assumes that malicious activity can be present within the system and needs to be detected and corrected. Note that in this context, *outsider* refers to an attacker without system access, whereas *insider* refers to a user with system access (i.e., able to transmit legitimate messages), rather than the physical location of the attacker.

More specifically, **proactive security** refers to any kind of mechanism that enforces a security policy. This category includes mechanisms such as integrity and authenticity checks (e.g., verifying cryptographic signatures), access control mechanisms and many other systems. In C-ITSs, the typical approach to provide authenticity is through the use of a public key infrastructure (PKI) to distribute keys, which are used to generate cryptographic signatures on messages. All unauthorized entities are excluded from the system, because their messages do not contain valid signatures. The state of the art for such PKIs is discussed in detail in Section 2.1.2. This creates a trusted perimeter that encompasses all authorized entities, requiring potential attackers to possess valid credentials to access the system, reducing the number of attack vectors. However, if an attacker compromises key material or otherwise manipulates the messages that are transmitted, attacks can still be successful.

Therefore, proactive mechanisms have to be complemented by **reactive security**. Reactive security consists of a *detection* and *reaction* step, where attacks that are not prevented by proactive security can be stopped. Detection includes a host of different systems, including those that analyze system behavior or system state to detect attacks and failures. These mechanisms can take many forms; for example, a detection mechanism could locate anomalous behavior that indicates an attack or fault. However, it is also possible to detect known attacks, as is commonly done in traditional corporate IT using, e.g., signature-based virus scanners, which detect a database of known malicious binaries. In the reaction step, the system response can vary from the revocation of a certificate to an instantaneous blocking or dropping of messages. For example, the program `fail2ban` is a commonly deployed tool against SSH password cracking attempts, which blocks login requests from the same source after multiple failed password attempts. In this thesis, we focus on the detection step in reactive security.

1.1.1 Intrusion vs. Misbehavior Detection

Reactive security mechanisms have a long history in traditional networks, where both intrusion detection systems (IDSs) and intrusion prevention systems (IPSs) are used. These are actively researched and many surveys on the topics exist [7, 11, 14, 34]. A general classification is through the method of detection: either by recognizing known attack patterns (signature-based), detection of anomalous behavior (anomaly-based), or detection of specification deviations (specification-based). Each of these approaches can be viewed as a classifier of traffic as either benign or malicious. The fundamental difference is that misbehavior detection instead classifies messages as being either correct or incorrect. Correctness is defined by whether the message contents correspond to real-world events, rather than by the presence of malicious intent. This means that misbehavior detection also aims to detect erroneous messages or the lack of messages, not just malicious messages.

In other words, misbehavior detection refers to the recognition of invalid application behavior, including the detection of attacks on data correctness. This requires that valid application behavior is defined, which in our work is primarily based on (application) semantics that exist beyond the messages that is analyzed. CPSs and C-ITSs are prime examples of systems that are suitable for such analysis, because physical laws and processes define valid application behavior. By validating observed messages against the baseline defined by these physical laws, it can be determined that the messages are invalid, i.e., the application is behaving incorrectly. In most work, this misbehavior is then attributed to malicious actors (i.e., attackers that attempt to inject malicious messages); in our work, we are primarily concerned with *data correctness* as opposed to attribution of the misbehavior.

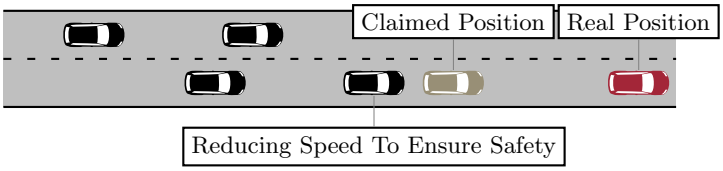


Figure 1.1: An example of a position falsification attack: the attacker claims a false position near a benign vehicle to cause it to reduce speed.

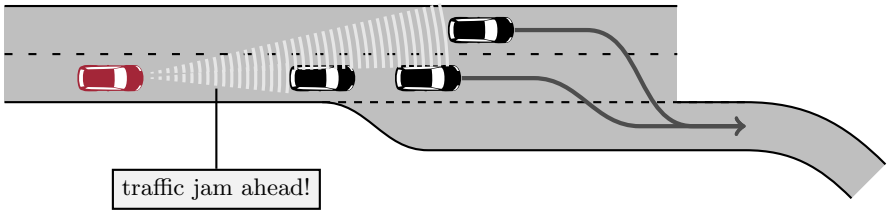


Figure 1.2: An example of a false event notification message: an attacker warns of a traffic jam ahead of the target vehicles, causing them to change routes.

1.1.2 Examples of Misbehavior

Because of the envisioned large-scale deployment and the direct interaction with the physical world, it is likely that intelligent transportation systems will attract a number of attackers. Many different types of misbehavior and attacks are conceivable, and we will provide an in-depth discussion and categorization in Section 2.2. For now, consider two example attacks to demonstrate the range of possibilities.

Figure 1.1 shows how an attacker might disrupt traffic by claiming false positions. Standards foresee that periodic message exchanges are used to enable each vehicle to be aware of the vehicles around it, which is subsequently used for a broad set of applications, including safety and driving comfort applications. In this example, the malicious vehicle claiming a position near another vehicle, with the goal of triggering some response from that vehicle, causing it to reduce speed. By repeating this process multiple times, an attacker can disrupt traffic on the entire road. This attack requires quite a sophisticated attack, since the false position claims are contained in the standardized periodic messages, which must contain valid cryptographic signatures to be accepted by the target vehicle. This thesis will discuss a variety of different misbehavior detection mechanisms to detect this type of attack in Chapter 6. This specific example can be detected by analyzing the movement pattern of the attacking vehicle, since different beacons will contain positions near different benign vehicles.

Figure 1.2 demonstrates a completely different type of attack. Here, the attacker uses knowledge of the protocol semantics to create a message that informs the vehicles ahead of an (non-existent) traffic jam. Receiving vehicles will use this message for routing decisions and might take an alternate route. As a result, there might be less traffic on the original road, which is a direct benefit for the attacker. The main harm caused in this case is that vehicles might slow down when approaching the non-existent traffic jam location, maybe creating a real traffic jam. In addition, the traffic participants might get confused in case they notice that there is no traffic jam on the original road. As a result the user acceptance of the system is lowered. In this example, the attack may be detected by observing that the notification is not coming from the direction of the claimed event, but rather from the other direction.

If we consider safety applications or automated driving [135], it is conceivable that attacks on intelligent transportation systems can even lead to accidents, possibly threatening the lives of passengers. For instance, given a set of vehicles that are using a platoon with reduced relative safety distance, an attacker could inject a fake message that indicates one of these vehicles is breaking, even though this is not the case. In such a situation, vehicles will immediately respond and in the worst case actually cause collisions to avoid the non-existent breaking behavior claimed by the attacker. This type of attack has been studied for its effectiveness in recent works [123], but suitable countermeasures require the ability to detect false information [159]. This is exactly what misbehavior detection is designed to do.

1.2 Existing Solutions

Misbehavior detection has been studied in the context of VANETs for some time [156, 164]. As part of this thesis we conduct an extensive literature survey describing the progress made in the first decade of vehicular communication research. This includes both misbehavior detection and the overall system model for C-ITS, which has developed significantly as research as progressed. Basic communication in these networks is done through an ad-hoc network that is created by periodic transmissions (referred to as beacon messages). More recent work has also considered the use of multi-hop messages in this network; all received information is stored by vehicles in their local dynamic map (LDM) [67, 99, 104]. This information is used to provide a variety of safety and driving comfort applications, as well as traffic information services. In recent years, the rise of partially autonomous vehicles have also become a potential application, enabling these vehicles to drive even more safely and efficiently.

Parallel to the developments of the C-ITS itself, a significant amount of research has been done to investigate the security implications of these systems. It became clear in the early stages of this work that in contrast to existing IT security solutions, the primary focus would be on availability and integrity, while confidentiality is significantly less important. This is because most of the applications necessarily require access to information from all the surrounding vehicles. The focus of security work has thus been to provide scalable solutions that provide integrity at a relatively low cost. Early proposals use public key infrastructures (PKIs) to issue certificates to vehicles; the associated private keys can be used to sign messages [18, 91]. To allow any vehicle to validate any received message, various proposals for *broadcast authentication* were made. Current standards require that certificates be attached to the message, in addition to the generated cryptographic signature.

Attaching certificates to every message makes it easy to track vehicles, raising significant privacy concerns [28, 32, 134]. To alleviate these concerns, most current-day standards foresee pseudonym certificates, which are typically certificates with a very short validity period (on the order of minutes) that do not contain identifiable information. This allows vehicles to authentically communicate with their neighboring vehicles without revealing their identity. To prevent advanced tracking techniques based on message contents, additional mechanisms such as mix zones and silent periods have been defined in the literature. These mechanisms increase the difficulty of tracking, but they require that a vehicle can use multiple pseudonyms in parallel or stop communicating for a short time. Since the latter is impractical for safety applications, multiple pseudonyms could be used in parallel, a feature that can potentially be abused by attackers, referred to as Sybil attacks [9]. For example, an attacker could use multiple pseudonyms to artificially inflate the amount of vehicles observed nearby.

Note that so far we have discussed proactive security mechanisms to prevent attacks from misbehaving in these networks. However, it is quite likely that at least some amount of vehicles will be compromised at some point during their lifetime. There are multiple avenues available for this compromise [24]. It is likely that extraction of key material from the vehicles will eventually be possible for some devices, which would enable attacker to sign arbitrary messages. Another possibility is a large-scale malware attack due to a software vulnerability in the vehicle; this enables generic attacks from a set of vehicles, even if the key material itself is still safe. Other possibilities include targeted attacks on vehicle-internal communication systems that cause the on-board communication device to transmit incorrect messages. To detect these types of attacks, misbehavior detection has long been seen as an essential component of the C-ITS [164, 170].

There are a broad variety of mechanisms available for misbehavior detection, which can be classified along many axes. One of these axes is the distinction between node- and data-centric detection mechanisms [102]. Node-centric detection mechanisms use concepts such as trust and reputation to determine the trustworthiness of senders, which is typically associated with the identity or pseudonym used by that sender. This can be based on previously received messages [51], voting [27], or it can be done through reporting [25] (i.e., a vehicle claims that another sender is malicious). Data-centric detection mechanisms, on the other hand, fundamentally analyze the trustworthiness of received data, rather than the nodes transmitting the data. These decisions are based on additional information, such as the expected transmission range, the speed of light, the maximum speed of a vehicle, or any number of other physical models. Data-centric detection mechanisms vary from simplistic plausibility checks [30] to complex behavioral modeling, using e.g., traffic flow to predict density and behavior [149]. There is also a broad set of detection mechanisms aimed at the above-mentioned Sybil attacks specifically [21, 90].

A number of authors have previously proposed frameworks to combine the results from different

detection mechanisms. One of the earliest such works is that by Golle et al. [12], who propose a data management framework that uses hypothesis testing as a basis for detection. The framework requires a number of simplistic assumptions, but the basic idea is that all data is stored in a database, and the framework searches for the hypothesis with the fewest number of messages labeled as malicious. Unfortunately, few details are provided with regards to the implementation of such a system, and the system itself lacks protection against Sybil attacks. A later work by Lo & Tsai [23] proposes a plausibility validation network, which basically combines a number of different plausibility checks. Using a rule database, the authors basically present a firewall-like approach that sets hard thresholds for correct messages; all incorrect messages are discarded. Compared to other work, the proposed rules are quite simplistic, and the lack of an evaluation of their performance makes it difficult to assess the impact of individual rules. Beyond these early proposals, several authors have also written a larger body of work with more extensive frameworks; these include VEBAS by Schmidt et al. [35], Stübing's PhD thesis and subsequent papers [73, 108], Raya's PhD thesis and subsequent work [40, 64], and Bißmeyer's PhD thesis [111]. We briefly discuss their weaknesses, which motivated our work; for a detailed discussion, refer to Chapter 2.

In VEBAS [35], the authors consider the fact that a large amount of detection mechanisms will be necessary to reliably detect most attacks. The idea of this framework is comparable to that of Lo & Tsai, combining a spectrum of plausibility checks into a local trust value. This is then combined with trust messages from other vehicles into an aggregated trust value. The local trust value is based on the addition of different detection mechanism outputs, which means that every time a new mechanism is added, the fusion approach must be revisited. If this is not done, then the combination of these results will be biased towards those attacks for which multiple detectors exist. There is also no clear way to deal with conflicting results in this framework.

The work in Stübing's thesis [108] made significant strides in the privacy-preserving detection of attacks on a specific application, namely tracking of nearby vehicles. This is done basically through a set of plausibility checks that must pass, after which the framework validates whether the vehicle is behaving as expected through a Kalman filter. This work has since been built upon by others with more advanced techniques, e.g., using an extended Kalman filter and vehicle-internal sensors [165]. The primary weaknesses of Stübing's work is the fixed combination of plausibility mechanisms, leading to a highly inflexible system with lots of manual configuration effort to add new mechanisms. This is especially problematic because such plausibility checks have to be valid in *any* situation that a vehicle will ever encounter.

Raya have designed a data-centric trust establishment framework in their thesis [40]. This work focuses primarily on the trust and revocation aspects of the framework, using game theory as a basis to show the limits of trust schemes. To establish trust, a variety of different tools are examined, including Dempster-Shafer theory and Bayesian inference. These reasoning tools enable a complex system that generates reliable trust, but the evaluation of this work relies primarily on whether an attacker is eventually revoked. In our work, we focus more on the timeliness factor, evaluating the framework's capability to give reliability estimates immediately after a message has been processed.

Finally, Bißmeyer's thesis [111] describes a specific misbehavior detection mechanism and discusses how this mechanism can be used for both local and global detection. Their work introduces the overlapping position detector, which combined with the work from Stübing is claimed to be a reliable detection scheme for position falsification attacks. This detection is transferred to a global misbehavior authority, in addition to local operation. Although this transfer to a global authority is useful, the evaluation methodology of their work is weak; neither data nor source code is available for any of their studies. We unsuccessfully attempted to replicate their work through re-implementation; we suspect the issues encountered are due to the same timeliness factor discussed in the context of Raya's work.

In summary, this thesis aims to address the weaknesses present in existing work. Some of these weaknesses arose through recent developments in the field of C-ITS, resulting in shifting requirements that invalidated previously reasonable assumptions. Others arise from the fact that designing individual mechanisms for misbehavior detection is simply quite hard, because there are many different attacks. These attacks are highly situation-dependent, and thus building a general classifier that splits all messages into legitimate and incorrect is unlikely to work. For example, on a highway, a beacon message with a speed of 0 km/h is legitimate when a traffic jam exists, but not under normal circumstances. In essence, we propose a flexible fusion approach that can encompass many mechanisms, each of which can be tuned to detect a specific attack; our fusion then enables accurate detection based on those results.

1.3 Problem Statement

Reliable vehicular communication is likely to be a key factor for safe mobility in the future. This has led to a surge of new research, investigating vehicular communication and its security in the new context of (partially) autonomous vehicles. Through the addition of communication, cooperative intelligent transport system (C-ITS) are born, raising new challenges for the defensive security community: how can connected autonomous vehicles be protected from malicious messages?

Adopting existing work that detects misbehavior in VANETs does not work for several reasons. First, most existing work on misbehavior detection requires a significant time frame before reliable decisions can be made. This is both due to the way detection mechanisms are designed, and because trust is considered a significant factor in these systems. Because trust is built over time, this causes inevitable delays. Secondly, existing solutions that detect misbehavior in VANETs have shown to be tied to specific attacks, and thus a combination of these solutions is required. In existing work, a static predefined combination of these results is used to detect attacks, which results in an inflexible system that can be potentially exploited by an attacker. It also makes it difficult to add new detection mechanisms that detect previously unknown attacks, potentially requiring a re-configuration of the entire system and all associated detection mechanisms. A holistic detection framework that enables fusion of different detection mechanisms without this issue of reconfiguration does not exist, to the best of our knowledge. In this thesis we focus on the development of such a framework, taking into account the new challenges raised by C-ITS and autonomous driving.

With increasing vehicular autonomy and thus control being taken away from the driver in some situations, it is no longer possible to rely on the driver to be the final responsible actor for decision making. Combined with increased speed and density promised by applications such as cooperative adaptive cruise control (CACC), where vehicles drive very close to each other to increase road utilization and save fuel, it is clear that security needs to step up. Many researchers are increasing the runtime performance of security mechanisms, bringing guaranteed integrity for the communication closer to reality. A direction that is only now experiencing a revival is the study of misbehavior detection for this communication: how can we guarantee correctness of the communication, rather than just the integrity of the messages? This thesis provides a main contribution to this question, as illustrated by the following research questions.

RQ 1. *How can detection performance of misbehavior detection in cooperative intelligent transport systems be improved, while also supporting the stronger requirements of autonomous driving?*

This is the primary research question of this thesis, asking how new security challenges raised by vehicular autonomy can be addressed, while at the same time improving detection performance overall. We divide this question into three sub-questions:

1. Can existing data-centric detection mechanisms be modified and improved to individually deal with noisy and uncertain information?
2. How can data fusion be applied to misbehavior detection mechanisms in a way that improves the overall detection performance?
3. What modifications to misbehavior detection mechanisms enable further improvements in detection performance?

We first provide a detailed overview of the state of the art in Chapter 2, explaining the differences with our work in detail. We then address the first question in Chapter 6, from which we conclude that meaningful overall improvements cannot be obtained by optimizing individual mechanisms. This is further confirmed by the simulation experiments conducted in Chapter 8, which show that individual mechanisms also imply a trade-off between false positives and false negatives. We discuss data fusion through subjective logic extensively in Chapter 5, and evaluate the resulting framework in terms of detection performance in Chapter 9. The results show that fusion can be used to meaningfully improve overall detection performance, especially when multiple attack types and orthogonal detection mechanisms are considered. However, we also show that detectors require modification to extract meaningful performance, confirming that applying data fusion is not a silver bullet. Approaches for these modifications are discussed, followed by concrete proposals and evaluation in Chapters 5, 6, and 9 respectively.

In order to answer our research question, we introduce two additional questions that were raised during the work done in this thesis, which are formulated as RQ 2 and RQ 3.

RQ 2. *How can data be stored, managed, and exchanged to support improved detection, inclusion of new detection mechanisms, and validation of detection?*

To allow the integration of novel detectors and support our proposals for improved detection, we require a useful representation of received data. As we developed such a representation in the early stages of this thesis, we realized that this representation can also be used to exchange detection results between actors. Inspired by this, we set out to provide a generic world model that enables any detection to take advantage of subjective logic, resulting in the following sub-questions:

1. What data management scheme enables reproducible detection results, while still enabling near real-time decision making required by cooperative autonomous vehicles?
2. How can subjective logic be adapted to fuse information from multiple sources at once, despite the non-associative definition of existing operators?
3. Can misbehavior detection mechanisms be designed to provide sufficient information, such that an issuer can reproduce the results and revoke the associated pseudonymous certificates?

We develop a world model representation of both received data and detection results, based on a graphical interpretation of subjective logic, which maintains historic information. Through this data management scheme, we can reproduce and redistribute detection results by sharing sub-graphs of this model. The resulting framework, Maat, is introduced in Chapter 3, while the details and trade-offs in data management are described in Chapter 4. To actually implement the fusion improvements we aim for, we also needed to extend the foundations of subjective logic. Chapter 5 describes these extensions, including multi-source extensions of several operators and a description of the use of information from unreliable sources. Our third sub-question provides an outlook on how Maat can be used to improve the overall trustworthiness of the C-ITS by enabling revocation of malicious actors. We do not provide a specific revocation scheme, but rather explain how results achieved in local instances of Maat can be redistributed in Chapter 6.

RQ 3. *How can we support the scientific development and analysis of misbehavior detection in cooperative intelligent transport systems?*

In order to draw scientifically rigorous conclusions from our results, a significant portion of this thesis addresses methodological questions. The new generation of misbehavior detection research addresses cooperative intelligent transport systems, which brings with it new requirements and therefore methodological standards and metrics. In this thesis, we place previous work on firmer methodological grounding and describe new metrics for the analysis of applications and detection performance. This process is sub-divided into three sub-questions:

1. Which evaluation methodology is suitable for the evaluation of misbehavior detection in cooperative intelligent transport systems?
2. What metrics can be designed that accurately reflect the real-world performance of cooperative intelligent transport systems, and which of these metrics are suitable for misbehavior detection?

The first two sub-questions are addressed in detail in Chapter 7, discussing in particular the subtleties involved in the choice of gold standard. For C-ITSs, we require a stricter gold standard, where the detection mechanism must detect all attacker messages, rather than just recognizing it within a certain time period (as was commonly done by earlier work). Because misbehavior detection is always a trade-off between false positives and false negatives, we also discuss metrics that can measure the implications of this trade-off. This results in a discussion of application-specific metrics in the context of CACC, where we the impact of attacks through reduced efficiency and collision impact. We also describe a generic metric to assess the dispersion of false positives and false negatives over a fleet of vehicles. Low dispersion (i.e., errors are concentrated in specific vehicles) implies that detector performance depends on the relationship between attacker and victim. This indicates that the exchange of reproducible detection results can enable a central authority to eventually revoke malicious actors.

Chapter 2

Related Work

This chapter introduces the C-ITS system model, followed by a review of the misbehavior detection literature, followed by a detailed discussion of different frameworks that have been proposed for misbehavior detection. Since the term *framework* is used loosely in the literature, this chapter concentrates on proposals that are essentially combinations of different approaches. Some use it to refer to an array of detection mechanisms, aligned to detect a specific attack (or protect a specific use case), while others concentrate on protection against a wide variety of attacks. Our discussion of frameworks is restricted to proposals that are sufficiently comprehensive to be considered a framework (as opposed to a specific detection mechanism), and we highlight the differences between those proposals and ours in Section 2.4. A wider discussion of ideas introduced by a more diverse set of authors can be found in our survey article [142, 170]. Parts of this chapter are based on this work, although the discussion here is extended with a discussion of the differences between these frameworks and Maat, the proposed framework of this thesis.

2.1 C-ITS System Model

Here we briefly introduce the basic C-ITS system model, as well as the baseline security measures that are currently being standardized and a brief overview of security credential management and privacy considerations. Although research on proactive security in this area is still progressing, we consider this to be largely orthogonal to our work. This is because some attack vectors will persist regardless of the proactive security mechanisms that are active. We will end this section with an overview of attacks that exploit those attack vectors.

2.1.1 C-ITS overview

The C-ITS system consists primarily of vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication, which is proposed to be based on the IEEE 802.11p amendment, which specifies MAC and PHY layers for short-range communication between vehicles. Communication is performed at 5.9 GHz with a communication range that is typically between 100 and 500 meters, and which is highly dependent on the scenario and buildings in the vicinity. IEEE 802.11p has gained world-wide adoption as a basis on top of which standardization agencies are building C-ITS applications. For example, the European telecommunications standardization organization, ETSI, has a family of standards for C-ITSs, referred to as ITS-G5, which provides communication primitives. On top of these, safety and entertainment applications are being developed. Similar initiatives exist in the US in the IEEE 1609 family of standards, in conjunction with the society of automotive engineers (SAE).

In contrast to routing in classical IT systems such as the Internet, multicast and broadcast are the predominant networking patterns in C-ITS [36]. These patterns are much more suitable for the envisioned applications, which vary from lane change warnings to cooperative adaptive cruise control (CACC) and city-scale traffic flow optimization. CACC is an application that has gained significant attention in recent years, and is essentially an extension of adaptive cruise control (ACC), where the safety distance between vehicles is reduced. In CACC, vehicles periodically exchange position information to form a very tight formation that would normally be susceptible to collisions [143]. This is done using a coordinated

controller that takes inputs from sensors and messages to compute a behavior that causes the platoon to be stable. The leader of the platoon is typically responsible for the maintenance of the platoon, while all the follower vehicles accept that vehicles' overall goal for the platoon. Platooning is thus a form of partially autonomous driving that goes beyond the potential benefits from sensor-equipped vehicles. The data exchanged between the network participants, vehicles and infrastructure alike, share the property that they are relevant to all receivers. Therefore, addressing packets to specific vehicles does not make sense; instead, addressing refers to the local neighborhood (1-hop broadcast), specific regions (geocast) and infrastructure. This style of addressing allows the network to exploit the simple fact that wireless networks are a broadcast medium by nature.

These communication patterns are supported by two message types in the European model: the periodic cooperative awareness message (CAM) [68] and the event-driven decentralized environmental notification message (DENM) [48]. In the US, standardization foresees a two-part basic safety message (BSM) [41, 92] that roughly corresponds to the CAM and DENM. These message types make up the bulk of communication between vehicles; in Europe, additional message types are currently under standardization for specific applications, such as the signal phase and time (SPaT) and topology specification (TOPO) message types. Both the CAM and the first BSM part consist of core elements including position, heading, speed, steering wheel angle and vehicle size. Supplemental information can be added, such as vehicle role and status of vehicle lights. CACC applications can technically work with just CAM/BSM.1 information, but most proposals include acceleration information into beacon messages for better performance.

The second BSM part is only transmitted when a specific event occurs, similar to DENMs, but is packaged within the next periodic BSM instead of being transmitted separately. This second BSM part is also single-hop; unlike the European model, which specifically foresees DENMs as multi-hop messages, events are transferred through 1-hop BSMs only. DENMs are event triggered messages, which are designed to warn about specific events, such as traffic jams, emergency breaking, an approaching emergency vehicle, or road construction. DENMs are usually relevant for specific geographical area, and can be forwarded there over multiple hops. A geobroadcast protocol is specified [69], which first forwards messages to the designated target region and then broadcasts the message within that region.

Although V2V communication using CAMs and DENMs is able to support a large number of use cases, some use cases require V2I communication. Infrastructure can help to increase communication range during the initial deployment phase, especially in critical areas such as intersections. Another important feature of infrastructure is that it enables communication with back-end infrastructure and the general Internet. This is useful for non-safety application classes, such as infotainment services, over-the-air software updates and security credential updates. RSUs can also be used to enable specific applications, such as traffic and fuel management through SPaT messages[1]. However, large-scale deployment of roadside units is considered unlikely due to the estimated deployment and operation costs of 3,000–5,000 US dollars per RSU [26].

Complementary to RSUs, cellular communication can be used to provide back-end and Internet connectivity, making use of the fact that broadband cellular communication services like long-term evolution (LTE) have already been deployed on a large scale. However, the necessary developments to support a large fleet of vehicles in addition to regular cell phones have not yet been made, despite recent strides in this area [175]. One of the major open question is related to the business model; thus many assume that cellular support will likely remain restricted to more expensive vehicles, at least as long as cellular usage still incurs considerable fees. New proposals from the area of 5G suggest that an ad-hoc-like mode dedicated for inter-vehicle communication will also be available in the form of LTE-V, also referred to as C-V2X, which is inspired by device-to-device communication (D2D) [163]. This approach has significant backing from industry, but lacks fundamental privacy guarantees in the same way that standard cellular communication does. Independent of C-V2X, some have proposed the use of a heterogeneous network in recent years, using both IEEE 802.11p-based and cellular communication [151]. This approach can take advantage of both networks, but also presents the challenge of efficiently managing the available channels in a decentralized way [140]. Both heterogeneous networks and pure cellular proposals are still in research phases, and will likely face many of the same security challenges as IEEE 802.11p. Although we focus primarily on the IEEE 802.11p setting, we expect the results of many schemes to transfer to C-V2X with limited modifications.

Beyond the communication model under standardization, some authors propose that clustering might

[1] These messages inform incoming vehicles when the traffic light will turn green, allowing more efficient fuel management.

be a viable approach to organize the communication. Current standards do not foresee this approach because it has significant disadvantages in terms of communication overhead to create and maintain clusters [42]. These clusters are typically created and maintained in an ad-hoc fashion, to facilitate deployment, which requires vehicles to agree on one or more cluster heads that manage a cluster. The disadvantage of this approach is that it requires a fundamental inequality between different vehicles to take advantage of clustering, i.e., the cluster heads must have an authoritative status. It is also unclear how well clustering can perform, which is highly dependent on the dynamicity of the network topology. For safety applications in particular, adding additional delays through clustering setup processes significantly increases the risk that essential information is communicated in time. In addition to these concerns, this network model faces security challenges that are widely different from the standard C-ITS communication models. Although it is realistic that the results in this thesis can at least partially be adapted for clustering, we do not address clustering explicitly in this thesis for the aforementioned reasons.

2.1.2 C-ITS Security

In order to mitigate possible attack vectors, security features are actively considered by both academia and industry. Standardization agencies have developed an initial standard for practical implementation, which serves as the basis for a lot of work in C-ITS security. Unlike traditional networks, though, confidentiality is not considered a major requirement, because most applications inherently rely on the data exchanged between vehicles. The main focus is instead to ensure integrity and authenticity, for which standardization employs proactive security mechanisms. The basic approach is to set up a PKI and provide each vehicle with an asymmetric key pair and a certificate. The certificate contains the public key alongside a number of C-ITS specific attributes (e.g., vehicle type, vehicle dimensions, license plate number) and is signed by the key issuing authority [24, 28, 32]. This serves as a long-term identifier of a vehicle that confirms it as a valid participant in the C-ITS. We first discuss the basic application of these certificates for C-ITS security, before discussing proposals for credential management in detail in Section 2.1.3.

The ETSI TS 103 097 standard [99] in Europe and the IEEE 1609.2 standard [104] in the US specify how certificates should be used to achieve integrity and authenticity. The basic idea is that each outgoing message is signed using the sender's secret key. The signature and certificate are both attached to the message to enable broadcast authentication, i.e., each receiver can check message authenticity without further message exchanges. For efficiency reasons, some authors have proposed certificates could be omitted periodically to save bandwidth [91], but this introduces additional delays beyond those already present due to time needed for verification. In both European and American standards, the cryptographic algorithm is the elliptic curve digital signature algorithm (ECDSA), using the p-256 curve standardized by NIST [106]. This signature and certificate validation process provides sender authenticity and message integrity, protecting against attackers that transmit messages using commodity hardware without key material.

While signature and certificates effectively thwart most attacks using commodity hardware without key material, they do not provide any guarantees on message correctness. Because of the expected wide deployment of C-ITS and the long lifetime of vehicles, it is likely that attackers either physically own a vehicle or are otherwise able to extract key material from (old) communication units. If key material is stored on a regular storage medium (e.g., hard disk or flash memory), attackers with physical access to a car could easily extract it, transfer it to other devices, and create arbitrary messages with valid signatures. Some do propose to use trusted hardware, usually in the form of a hardware security module, to protect key material [24, 28]. Here, the idea is to store (secret) key material within a tamper resistant component that is protected against outside access. If a message needs to be signed, it can be forwarded to the trusted device and the signature is returned. Thus, the key material never leaves the trusted hardware, making it much harder for an attacker to extract secret keys and subsequently sign arbitrary messages. As in normal PKIs, revocation can be employed to mitigate this issue, but this raises new questions on how key material abuse can be detected in existing vehicles.

Note that if key material is kept in trusted hardware, the application code is still untrusted: it is likely that attackers are able to manipulate the software in order to create arbitrary messages. These messages will then be signed, because the signing functionality running on trusted hardware has no way to tell whether messages have been manipulated [18]. The alternative would be to build an architecture

in which all hardware is trusted, but this also has many disadvantages. First, running all software on trusted hardware will greatly increase cost, because more powerful trusted hardware is needed. In addition, if all applications need to be manually certified and deployed on trusted hardware within the vehicle, over-the-air updates are more complicated and the deployment process of new software is slowed down. Even if attackers are not able to modify the software, they may be able to modify sensor readings, which lead to modified messages as a result. To alter sensor readings, attackers can either inject false readings in the controller area network (CAN) bus or directly modify sensor hardware. Therefore, it is likely that software running on the vehicle can be controlled by a capable attacker, despite trusted hardware. This requires us to consider correctly signed messages with invalid contents circulating in the network.

2.1.3 Certificate Management & Privacy

If vehicles attach their long term certificates to all outgoing messages, they can be tracked using a trace of received messages with attached location information. For this reason, long-term certificates can be replaced by short-term identifiers, referred to as pseudonyms. These pseudonyms make it harder to collect location traces of specific vehicles [28, 32, 134], while still providing the same authenticity guarantees. Pseudonyms significantly improve privacy, although their efficacy in the absence of protocol-level changes may be limited [62], due to the fact that message contents allow linkability. While pseudonyms enhance privacy, their parallel use enables potential attacks, often referred to as Sybil attacks [9], which is a class of attacks where multiple identities are used by one actor (see Section 2.2.1). In this section we first present a discussion of some proposals for certificate management.

A standard PKI only foresees the model as discussed above: each entity has a valid certificate that identifies the entity, and authenticity entails identification. For privacy reasons, many proposals have been made to issue a set of pseudonyms, which is typically linkable under certain conditions to the long-term identifier of the entity [134]. The creation, distribution, management and revocation of these pseudonyms present a trade-off between privacy, security and performance. Two major proposals are now in the initial stages of deployment, which analogous to standardization of communication can also be seen as a European and a US proposal. In the US proposal, Brecht et al. [168] propose the use of *butterfly keys* to provide a cryptographic link between different certificates. In this model, pseudonyms are issued in bulk, revocation is done by revealing the butterfly key, and a misbehavior authority interacts with the pseudonym issuing entities to detect attacks in the back-end. Recent European proposals, such as that suggested by Khodaei et al. [173], take an on-demand approach to pseudonym issuing, and claim to have more protection against Sybil attacks. Such protection can be provided by limiting the validity of pseudonyms, which is a trade-off against performance in the US model (where many pseudonyms would be wasted due to batch issuing).

In general, these pseudonym issuance systems provide a partial framework that in turn influences the suitability of certain detection schemes. At the time of writing, it is still not clear which exact scheme will be implemented world-wide, or whether disparate systems will be used in different parts of the world. However, the constraints posed by the credential management system impacts potential detection mechanisms, primarily through how easily Sybil attacks can be executed. This aspect of pseudonym management is addressed in our literature systematization, as discussed in Section 2.3.3.

2.1.4 Attacks on C-ITS

Here we briefly review a number of proposed attacks on the C-ITS, particularly those that can still be performed within the scope of existing security mechanisms as described in the previous subsections. These are divided along the lines of the cyber and physical in a cyber-physical system; in this thesis we primarily focus on the cyber component. A detailed discussion of adversary models can be found in the work by Papadimitratos et al. [19]. For attacks on the cyber layer, the attackers of primary interest for C-ITS have turned out to be internal active adversaries with bounded presence. For the physical layer, a natural first choice is the input-controlling adversary discussed by Papadimitratos et al. [19], although this model may be too strong for some specific sensor types.

Cyber attacks

The space of possible attacks on computer systems is vast; we concentrate on attacks that are not easily avoidable through security mechanisms that are already implemented. As there are many survey articles discussing attacks [136, 156, 164], we here provide a review of the important types and refer interested readers to those articles for a detailed review. The attacks we discuss here are jamming attacks, data injection attacks, replay attacks, routing attacks and Sybil attacks.

In a jamming attack, the attacker disrupts communication from or to specific nodes by transmitting a constant or targeted burst communication to disrupt communication between nodes. This essentially means that the attacker can decide which messages will arrive, and which do not. This attack is commonly discussed in CACC and can be executed even by low-power devices such as unmanned aerial vehicles (UAVs) [150].

Replay attacks, on the other hand, refer to an attacker re-transmitting received messages. Standard proactive mechanisms provide some protection against replay attacks by including a time stamp in every message. In C-ITSs, a relatively reliable source of timing is available in the form of a global navigation satellite system (GNSS), which also provides high-accuracy time synchronization [147]. Attacks on GPS, currently the primary GNSS assumed by many works, can be modeled as input-controlling adversaries, and are relatively well-studied. Many effective countermeasures have been designed that protect against such attacks [147], and thus we consider these attacks to be out of scope. However, in multi-hop communication it is conceivable that these attacks can affect throughput.

Replay attacks in multi-hop communication are closely linked to routing misbehavior, a well-studied class of attacks that aims to disrupt network performance. This category of attacks originates from mobile ad-hoc networks (MANETs), and both attacks and defense mechanisms have since been transferred to vehicular ad-hoc networks (VANETs) and other types of ad-hoc networks. Basically, the attacker deviates from the routing protocol to cause messages to be lost or be routed through a specific node. Unlike most attacks we study, routing attacks do not directly affect safety, and thus we consider these attacks to be of limited importance. Note also that C-ITSs includes infrastructure; if available, using infrastructure by default for long-distance communication significantly reduces the impact of routing misbehavior.

Another important class of attacks is that of data injection; in these kinds of attacks, an attacker claims information that contradicts with real-world information. The purpose of such an attack can be disrupting road traffic, or even triggering a collision. For example, attackers may falsely claim that a road is blocked to have the road for themselves, or they may claim a vehicle is directly in front of a victim, causing the victim to trigger the breaks. As suggested by these examples, attacks have widely different goals, with varying time-sensitivity. Because of the variety of attacks, many different mechanisms to detect them have been proposed in the literature, and a large chunk of misbehavior detection is directly related to these attacks. Authors have shown [123, 159] that CACC is particularly vulnerable to these attacks, and being able to detect them is essential for the design of secure control algorithms.

Sybil attacks [9] are a specific type of attack associated with the pseudonym discussion above. In the simplest case, an attacker here uses multiple identities to achieve a specific goal. In C-ITSs, this goal is often related to the application, similar to data injection attacks: an attacker may use multiple pseudonyms to make it appear as though the road is full, even though it is not. Closely related to this are various types of attacks on specific detection mechanisms, where the attacker uses the identities to boost the reputation of specific vehicles, or conversely reduce the reputation of legitimate vehicles (*bad mouthing attacks*). A detailed discussion of reputation systems and attacks on them was provided by Koutrouli et al. [133]. It is important to remark that Sybil attacks can be performed not just on reputation systems, although this is a widely-studied case; we return to this topic in our discussion of the attacker model (section 2.2.1).

Physical attacks

Apart from the cyber component, there are also many attacks possible on the physical processes surrounding the vehicle itself. Recent work on the security of CANs [54, 74], which are the primary type of bus used in current vehicles, has shown that they are vulnerable to attacks. So far, this work has primarily focused on attacking a vehicle from the outside; however, an attacker attempting to attack the C-ITS could similarly exploit their physical access to the bus. Such attacks allow an attacker to force the on board unit to transmit arbitrary messages that conform to the network specification without needing to compromise the hardware security module. This situation reflects the input-controlling adversary of

Papadimitratos et al. [19], and it leads us to conclude that attacks within a specific vehicle are likely to lead to false messages being disseminated in the C-ITS.

Some authors have also suggested that physical attacks on specific vehicles may be possible. This works by directing the attack towards the sensors themselves [135]. A simple example of such an attack would be taking a laser pointer and disrupting the cameras of a nearby vehicle. We primarily focus on attacks in the cyber space; however, we feel it is important to also consider that physical sensors are not necessarily immune to attacks.

2.2 Misbehavior in C-ITS

In this section, we will go into more detail concerning misbehavior in C-ITSs. In particular, we discuss the attacker model and different types of realistic attackers; this will be the basis for the remainder of the thesis. We provide a detailed review of existing misbehavior detection mechanisms in Chapter 2.

2.2.1 Misbehavior Definition and Attacker Model

In this subsection, we will treat the definition of misbehavior and the attacker model associated with it in the context of C-ITSs. This definition is based on the notion of misbehavior discussed above, but integrates it within the state of the art of C-ITSs.

Defining Misbehavior

In literature, there are many different definitions of misbehavior [18, 24, 25, 43, 78], often implicitly defined by the attacker model rather than explicitly being stated alongside it. Misbehavior is a broad term; its origins are somewhat unclear, but it is commonly used for ad-hoc networks when discussing specific attacks that are executed by the participating nodes, as opposed to intrusions or attacks. Our definition of the term covers not only attackers and malicious participants, but also faulty nodes. *Misbehaving nodes* are thus any node that transmits erroneous data that it should not transmit when the hard- and software are behaving as expected. We argue that a misbehaving node is the type of node we should detect. However, the literature often distinguishes [22, 23, 28, 31] between *faulty node* and *malicious node*, which are defined as follows:

Faulty nodes are those participants in the network that produce incorrect or inaccurate data without malicious intent. Faulty nodes are usually related to faults in sensors, either because the sensor was damaged or because of errors caused by variance in the sensor. Examples include a heat sensor that is malfunctioning, causing the node to transmit the current temperature to be -50 degrees Celcius, and the error caused by a reading produced by a GPS device, which can cause vehicles to transmit an erroneous position.

Malicious nodes or attacker nodes are those nodes that are transmitting erroneous messages with malicious intent. Such messages may also be referred to as deceptive messages, and the attackers as deceptive attacks. These nodes are our main target for detection, as they represent a direct threat to the integrity of the data exchanged in the network. These nodes can actively attempt to avoid detection, or subvert other nodes in the network to transmit their erroneous messages. This definition includes denial of service attacks and exclusion of messages that should be sent (e.g., attacks on routing). The goal, source and means of these nodes may vary greatly, as discussed below.

Note that these definitions are not consistently used throughout the literature; in our work, we use misbehavior detection to refer to the detection of both faulty and malicious nodes. Unlike traditional intrusion detection, misbehavior detection attempts to detect incorrect packets, rather than detecting malicious packets. There are many good reasons to avoid a focus on content in a generic networking setting: unlike in C-ITSs, such networks see a lot of encrypted traffic and a large variety of application layer protocols. In C-ITSs, we have relatively few protocols, and the public nature of transmitted data implies that content is rarely encrypted. In addition, C-ITSs lack a clear system boundary, which is essential for effective IDS deployment. Misbehavior detection can take advantage of the public nature of data, and does not suffer from the lack of a system boundary, making it a potentially more suitable approach. This type of detection may not be feasible for normal networks, but the characteristics shared between C-ITSs and cyber-physical systems (CPSs) suggest some schemes could be used in both systems.

Attacker Model

We now review several attacker models that are used in the literature. The secondary purpose of this section is to discuss the challenges involved with the development of a good and universal attacker model for C-ITSs. Unlike most network scenarios, there is no universally accepted attacker model that is consistently used for C-ITSs; here we provide a brief overview of the most common assumptions.

The attacker model from [24], one of the seminal works on security in C-ITSs, presents the following four classification dimensions for attackers, and a variety of basic and sophisticated attacks. We already discussed the distinction between *insider* and *outsider*, i.e., whether or not the attacker possesses valid credentials. The motivation of the attacker is classified as either *rational*, where the attacker has a direct benefit from his attack, and *malicious*, where an attacker only seeks to disrupt or cause harm. The attacker may be *active* or *passive*, which considers whether the attacker can transmit messages or signals, or whether the attacker only eavesdrops on the channel. For this thesis, we seek to detect active attackers only, as the very goal of misbehavior detection is to determine whether certain messages or signals constitute undesirable behavior; eavesdropping is not misbehavior, in the sense that it cannot be detected. The final classification dimension is the scope of the attack, which may be *local* or *extended*. This distinction does not consider the amount of attacker-controlled nodes, but rather their *distribution* over the network. Local refers to one or more attackers distributed in a small region, such as a highway section between two intersections or a few neighboring intersections in an urban setting. The extended scope provides for a number of attacker-controlled nodes across a larger region.

Most attackers in this classification are instances of the standard Dolev-Yao attacker model [3] that is used for cryptographic protocols. However, the insider attackers that behave according to the protocol, but send false information are not covered in the attacker model. These can instead be modeled using an input-controlling adversary, as discussed in detail by Papadimitratos et al. [19]. In addition, as discussed by the authors [24], Sybil attacks play a significant role. In those attacks, attackers obtain multiple identities (multiple certified key-pairs) to circumvent reputation mechanisms. Many authors of detection mechanisms implicitly or explicitly limit the potential for Sybil attacks, meaning the attacker has possession of a small set (usually 1-3) of valid pseudonyms for each vehicle they control. As mentioned previously, recent proposals for PKIs include such considerations [173] to limit the available pseudonyms for this reason. Other proposals [168] suggest that such solutions are not cost-effective; in addition to this, there is a potential impact on privacy due to the potential of tracking [62].

One alternative means to address the challenge of Sybil attacks without violating privacy is the use of hardware security modules (HSMs), as mentioned by several seminal works on VANET security [24, 28, 32]. There have also been many proposals to implement such hardware-based solutions [66, 166]. However, such an approach faces many unsolved issues, which is why in this thesis we assume the PKI-based model, where the limited validity of pseudonyms is used to address the risk of Sybil attacks. These issues are primarily related to the input-controlling adversary model [19], where an attacker might attack the network by compromising a vehicle from within. Since the attacker has full control of the input of a vehicle in this case, the vehicle must be assumed to transmit false messages, unless the entire vehicle is part of the trust base. We argue that the latter assumption is unrealistic. Limited pseudonym validity can be achieved either at the certification level or within the hardware that uses the key material. We assume that this occurs at the certification level, since we expect that wide distribution of HSMs in vehicles is likely to lead to at least some of these devices being compromised (which would enable Sybil attacks).

One concern is that many authors [24] consider RSUs to be fully trusted. We remark that although this may be the case for connections leading outward through the RSUs, full trust cannot be assumed for the devices themselves. As the name implies, road-side units (RSUs) are deployed on the side of the road, which makes them susceptible to physical attacks like sensor tampering and differential power analysis, just like regular vehicles. Because RSUs are potentially more authoritative, they are potentially valuable targets if too much trust is placed on them.

On the other hand, the full Dolev-Yao model, with the extensions and limitations described above, is too strong to provide a meaningful level of security. Particularly, if the attacker is allowed to arbitrarily re-organize the packets received by each individual node in the network (as in the Dolev-Yao model), they can selectively deliver only those messages that confirm the view presented by the attacker. In reality, the attacker is also subject to the limitations of physics. For example, an attacker cannot arbitrarily re-arrange the reception of messages for each receiver in a wireless network. For this reason, the literature

specifies an important limitation compared to the Dolev-Yao model: an attacker must be a node in the network, restricted by the same physical properties that restrict a normal node. Although their transmission range may be increased through power control [176], attackers are considered to have a limited physical presence [32]. In addition, it seems reasonable to assume that attackers are constrained to fundamental physical limitations such as the speed of light. Similarly, an attacker cannot in general receive and jam a message at the same time without the risk that some other receiver also successfully receives the message. Another issue with the Dolev-Yao model is that it is not designed for the analysis of attacks on application semantics, i.e., message correctness is not considered, which is the primary focus of misbehavior detection.

In the literature, some authors have attempted to formalize these attacks and the effects on specific mechanisms using game theory [44, 57]. These formalizations often make an informal honest majority assumption, which assumes that a majority of the nodes (or sometimes keys or messages) is honest. The assumption is sometimes explicitly specified as being about a local neighborhood or the entire network. A local honest majority assumption is then the assumption that more than half of the nodes in the direct communication range of a vehicle is honest. This assumption is the most significant limitation of some classes of detection mechanisms, which we will discuss at length in our literature review in Chapter 2. Apart from this, we will rely on terminology introduced by earlier work [24, 29, 32, 65].

2.3 Literature Systematization

We now briefly discuss a systematization of the misbehavior detection literature along different lines. As this classification primarily provides the backdrop for the discussion of the related frameworks in the next section, we will avoid going into a detailed discussion of specific work. We classify mechanisms by three different variables: the position in our misbehavior taxonomy, the scope of detection, and the ability to deal with pseudonyms. Briefly summarized, our taxonomy classifies mechanisms by what the semantics of their output is (data- or node-centric), as well as whether they consider each vehicle in isolation (autonomous), or whether they analyze data from multiple sources at the same time (collaborative). The detection scope asks whether detection occurs within the vehicle (local), cooperatively between vehicles, or globally. Finally, the impact of pseudonyms is analyzed by examining what level of linkability is required for the mechanism to work correctly: full, explicit, implicit linkability or none at all.

2.3.1 Taxonomy of Misbehavior Detection

We have designed a taxonomy to categorize mechanisms, consisting of two classification aspects and four classes, as shown in Figure 2.1.

The first aspect we use for classification distinguishes between node-centric and data-centric mechanisms, which has regularly been used in the literature. For our purposes, node-centric mechanisms are defined as mechanisms that specifically evaluate the participants of the network, as opposed to the data exchanged within it. For example, they can verify the forwarding behavior of a node by analyzing packet frequencies, correctly formatted messages, and so on to decide on its trustworthiness. Similarly, trust establishment schemes that judge vehicles based on previous behavior or exchange of reputation information would be considered node-centric in our taxonomy. On the other hand, data-centric mechanisms evaluate the trustworthiness of the data itself, as opposed to the trust in nodes. Examples of data-centric mechanisms include a check on the plausibility of the data (e.g., is the claimed speed within a certain range) and the consistency of the data (e.g., does the claimed position overlap with that of another vehicle). Trust establishment schemes that are based on behavior typically use information provided by data-centric mechanisms, and thus there is some overlap in the authors working in these areas. However, we split the taxonomy this way because the basis for the decision is separate, and thus the ideas used in these two classes of detection mechanisms can be seen as orthogonal.

The second aspect we use to classify misbehavior detection mechanisms is the distinction between mechanisms that analyze messages from a single vehicle (autonomous) and mechanisms that attempt to deduce misbehavior from multiple vehicles (collaborative). This distinction is best illustrated using an example from a data-centric mechanism. A misbehavior detection mechanism can analyze a stream of messages from a single vehicle to detect suspicious behavior, or it can analyze all messages received in a time slot and detect messages that deviate from the majority. A significant advantage of autonomous detection is that the mechanism will function independent of any attackers that may be present, while

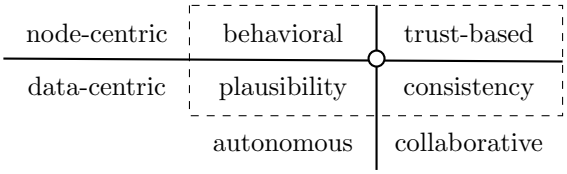


Figure 2.1: Taxonomy of misbehavior detection.

collaborative mechanisms rely on the fact that an honest majority exists. We note that autonomous detection may sometimes include the use of the vehicles' own sensors (e.g., when determining the approximate source of a radio transmission, or the relative position), which we still consider to be autonomous. Some consider this approach to be the only one that provides a reliable system. However, the challenge with autonomous detection mechanisms is that they are often imprecise, and cannot necessarily detect intelligent attacks that use knowledge of the detection algorithm. This imprecision is best exemplified by plausibility checks, such as a limit on the claimed speed that is accepted; an intelligent attack can always attack such checks by choosing values close to the limit. To avoid a large number of false positives, the limit on the accepted speed should be high enough to deal with outliers (e.g., speeding vehicles), but low enough to detect malicious behavior. If correctly employed, a set of plausibility checks bounds the message space available to the attacker, but also provides a space in which the attacker can choose messages that will be considered valid. This resulting bound can be used by employing collaborative mechanisms to provide a higher quality of detection than is possible using either approach individually. Indeed, the field is progressing towards these ideas, as evidenced by several detection mechanisms we discuss in our survey [142] and in our summary below. Having discussed these classifications, we now describe the resulting four classes from Figure 2.1.

Node-centric Misbehavior Detection

Behavioral The first branch of node-centric detection is behavioral. This type of detection exploits patterns in the behavior of specific nodes at a protocol level, where behavior is largely independent of physical processes. Information considered by these mechanisms includes the amount of messages transmitted by a node or the correctness of their format. A core aspect of behavioral mechanisms is that their analysis is focused on a node-basis and typically does not consider data semantics as done by data-centric mechanisms. An example of a behavioral misbehavior detection mechanism is the concept of a Watchdog [8] that was introduced for the security of routing in mobile ad-hoc networks. In a Watchdog, each node monitors the network, listening for retransmissions to verify that its neighbors, which are supposed to forward messages, actually do so.

Trust-based The other branch of node-centric detection is trust-based. Trust-based mechanisms exploit the fact that many nodes in the network are honest, and that infrastructure is available to remove malicious nodes. Trust-based detection includes reputation systems, which rate node behavior over time, but also voting schemes that allow vehicles to vote on the correctness of information. Trust-based detection can occur either locally or with infrastructural support. In the latter case, issues regarding privacy are a serious concern, which we address in Section 2.3.3. Trust-based detection often relies on input from other detection mechanisms to update the reputation of nodes in the network.

A core advantage of trust-based mechanisms is that they simplify the revocation process. However, the ephemeral nature of C-ITSs poses additional challenges to trust-based mechanisms, particularly when determining their initial trustworthiness, as well as the information to update it. Data-centric detection mechanisms or behavioral detection mechanisms may be used for this: the trust-based detection mechanism is then used to distinguish between legitimate and malicious behavior. Both reputation- and voting-based mechanisms have to deal with Sybil attacks [9], where an attacker abuses the reputation mechanism by creating multiple identities. These attacks can be used to either amplify an attack or exclude legitimate nodes from the network, and is one of the core research challenges for trust-based misbehavior detection, especially in combination with privacy requirements. Similarly, if an honest node suddenly starts misbehaving (e.g., due to a faulty sensor), this will not be detected quickly by reputation mechanisms.

Data-centric Misbehavior Detection

Consistency Consistency-based detection uses relations between packets, typically from multiple participants, to determine the trustworthiness of newly received data. For example, a consistency-based detection mechanism may consider a previously computed average speed of vehicles on a highway to judge newly received messages. Messages that deviate from the average are inconsistent with the known state and can thus be considered suspicious. Alternatively, pairwise comparison of messages from different vehicles is also considered consistency-based. Consistency-based detection has the advantage that only limited domain knowledge is required to design reasonable schemes. However, a local honest majority is often required to draw reliable conclusions: if several colluding attackers surround a victim, consistency might exclude information from legitimate vehicles.

Plausibility Plausibility-based detection uses a specific underlying model of data in order to verify if the transmitted information is consistent with this model. For example, plausibility of movement can be verified from two subsequent beacon messages by examining the distance traveled between them and comparing it to the speed in these messages. Plausibility-based typically allows for a very rudimentary but fast analysis of received packets. It is performed by considering packets from individual senders. The information in the packet is either tested against a prediction of the model, or the model is used to judge whether the information in the packet is a plausible next step according to the model. Because there is an underlying model, the mechanism can directly express the plausibility of the message in a probability, which can be input for further misbehavior detection mechanisms. The models used for plausibility vary from narrowly defined models like the laws of physics up to models that allow a wide range of variation, such as a model that predicts driver behavior. The narrowly defined models can be effectively used to filter incoming packets for "impossible" messages that can be discarded directly. A major advantage of plausibility-based detection is that the mechanisms are always applicable, even when an honest majority does not exist. A significant disadvantage is that a model of the underlying data is required, and the utility of the mechanism depends directly on how accurate this model is.

As is the case for node-centric misbehavior detection mechanisms, data-centric detection mechanisms have also been used in MANETs [100], which has seen continued developments as VANETs and C-ITSs have advanced. For example, Fiore et al. [100] present a set of data-centric detection mechanisms that also fall in the category of position verification. However, in this thesis, we concentrate on designs specific to VANETs and C-ITSs. The primary difference between these is that the application behavior and protocol designs are different. For example, Fiore et al. [100] discuss the use of a neighbor verification protocol that can be used for cooperative position verification. Such protocols can be complementary to the work done in the context of C-ITSs; for this thesis, we conservatively assume that such protocols are infeasible.

2.3.2 Scope of Detection

A second classification of detection mechanisms is illustrated in Figure 2.2, categorizing by the scope of detection: local, cooperative or global. Local detection refers to detection that checks internal consistency, and optionally the vehicles' sensors, as measures for correctness. Cooperative detection relies on collaboration between vehicles (and possibly some RSUs); note that this is distinct from the *collaborative* component in our taxonomy. Finally, global detection refers to detection that occurs at least partially with support from a back-end system. This terminology is used throughout the literature describing the different mechanisms, although some distinctions exist as to what constitutes a cooperative setting, relating to the number of attacker-controlled vehicles that are in a specific region.

Most detection mechanisms in the behavioral and plausibility classes operate purely locally, which makes them invariant to Sybil attacks. However, this also makes it difficult to identify the attacker, or even the attack, due to the low amount of available information. Some consistency-based mechanisms also operate locally, by comparing series of beacons from several vehicles, exposing them to Sybil attacks, in exchange for a much more fine-grained approach to detecting attacks.

On the other hand, there exist schemes that perform detection cooperatively: these are typically consistency- and trust-based detection mechanisms. These detection mechanisms often rely on an honest majority and exchange messages between participants to detect inconsistencies or untrustworthy participants. Particularly, trust-based detection is based on cooperation between nodes; most schemes that

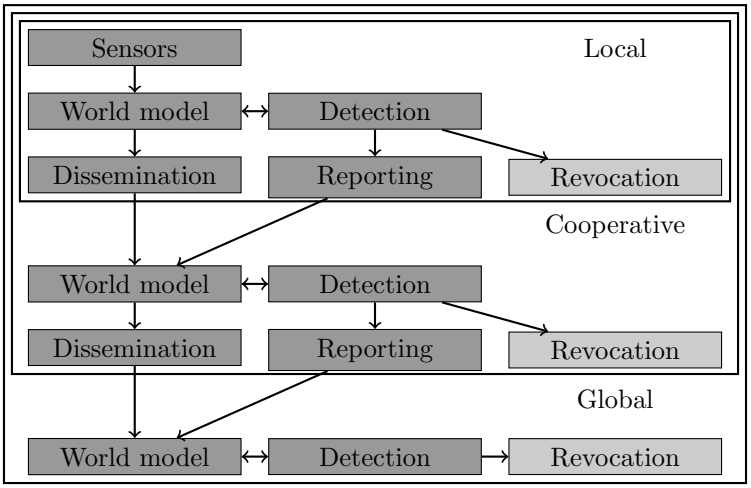


Figure 2.2: Different scopes at which detection mechanisms can operate.

operate as a reputation mechanism incorporate a mechanism to perform reporting and/or revocation, both cooperatively and globally, as this is typically a straight-forward extension of reputation. Although we primarily focus on detection, the ability to perform revocation implicitly within the vehicle is a useful property.

2.3.3 Impact of Pseudonyms

As mentioned previously, pseudonym systems can have an impact on the types of misbehavior detection that can be performed. In particular, pseudonym schemes aim to hinder linkability of different information items. Linkability is an important aspect of many models used for misbehavior detection, and sometimes even critical for safety applications. We propose four simple classes that we will use to classify detection mechanisms in our survey: full, explicit, implicit and no linkability, which are briefly explained below. For more details on the subtleties of pseudonyms in C-ITS, see Petit et al. [134].

Full linkability means that all messages transmitted by the same on-board unit (OBU) can be linked, i.e., there is no pseudonym change at all, and thus essentially no privacy. All possible misbehavior detection mechanisms can be realized at this level and some types of misbehavior detection even require this level of linkability. Examples of such schemes include most trust-based approaches [64, 76].

Explicit linkability refers to any type of linkability that allows direct access to an identity. This means that linking is technically possible; however, there are several ways to implement this. The issued certificate can either contain the *identity in an encrypted form*, a *direct mapping* from certificates to identity can be stored in the back-end, or this mapping may be split across multiple entities through *separation of duties*. Using each of these classes, it is technically possible to link pseudonyms, but this capability is limited by the pseudonym scheme.

Encrypting an identity in the certificate is an attractive approach, because there are cryptographic schemes that enable conditionally revealing this identity (e.g., PriPAYD [83], or double spending prevention in electronic payment systems). However, it is not clear whether this can also be applied for misbehavior detection, especially if revealing the identity should happen locally. This system also requires that detection mechanisms are standardized, and in most cases part of the certification infrastructure, which is an unrealistic assumption. Using a *back-end mapping* is a solid alternative to this approach, in terms of implementation effort. Here, each certification authority simply stores the relationship between pseudonyms and real identities, revealing malicious users as needed. This is particularly useful when combined with direct back-end connectivity, either through RSUs or another technology, such as universal mobile telecommunications system (UMTS) or LTE. However, this requires full trust in the back-end system, as it can revoke the privacy of the users. Another advantage is that a back-end could perform its own misbehavior detection, and exploit the significantly higher computational power available to it [89].

A middle way between these two types is *separation of duties*, which also stores the link between two pseudonyms, but spreads the information necessary to reveal a user across organizational boundaries. VToken [59] and Butterfly Keys [109] are different proposals that achieve this; overall it seems that industry and academia both prefer separation of duties over purely technical solutions, which require the cooperation of the revoked device (as is the case for PriPAYD, for example). The core advantage is that the separation of duties allows individual decisions on a case-by-case basis, if needed. However, for misbehavior detection, this means that linkability between pseudonyms cannot be achieved locally in the vehicle, but only on an organizational level. The advantage is that confirmed misbehavior can be acted upon immediately, and some mechanisms may be able to detect attacks by using a partial linking protocol to perform specific checks for messages that are suspicious if the pseudonyms would belong to the same identity (or vice-versa).

Implicit linkability or *inference* allows pseudonym linking even when no mechanism for direct identification or linking is available. There are three different sources of information available for this purpose: *certificates*, *message content* and *signal properties* during transmission. Using this information, partial or complete identities may be derived, although there is no guarantee that such a linking approach will always be able to identify every vehicle. For example, certificates typically contain attributes of a vehicle, such as length, height and color. These properties allow full identification of unusual vehicles on the road, such as a yellow limousine, but not the identification of several black vans of identical make. Although this is an example of linking based on certificate content, this type of limitation applies to all inference-based linking, and while it can be bounded by combining different types, it will have implications for misbehavior detection.

Secondly, implicit linkability does not fundamentally provide linkability through an identity, but through similarities in messages. Implicit linkability only provides links between messages, which will allow a receiving vehicle to create 'pseudo-identities' for groups of pseudonyms that are considered to be the same vehicle. This makes it more challenging to exchange information about particular vehicles with other participants, and can also make it difficult to verify misbehavior and revoke the misbehaving vehicle. Nevertheless, many misbehavior detectors will need to function under these assumptions, as they provide the most realistic environment in which individual vehicles have to perform misbehavior detection (as opposed to detectors in the back-end). Examples of inference-based systems include virtually all data-centric security mechanisms. For signal-based inference, the work by Guette & Ducourthal [21] and Xiao et al. [20] are good examples, while for inference on message content, a good example is the Kalman filter [61, 82]. Inference based on certificates is a relatively new idea that is sometimes captured within inference through messages, and is not commonly considered because it conflicts with the assumption that these certificates are intended to be pseudonymous or anonymous towards other participants. Therefore, they are typically assumed not to reveal this information, making such detection mechanisms obsolete.

No linkability is the idealized scenario in which it is impossible to determine whether two messages originate from the same or from distinct vehicles. Although this results in maximum privacy, such a scenario makes it nearly impossible to provide any working C-ITSs applications, let alone securing them. For example we can still attempt to perform misbehavior detection on individual messages. The goal of such an analysis is to detect unusual or inconsistent messages, which either contain obviously incorrect values: this includes unrealistic values (speeds of 500m/s) and values that do not match the perceived scenario (speeds of 50m/s in a traffic jam). The value of such detections is limited from a security point of view, but filtering malicious packets may still be beneficial for applications. Beyond that, not much more is possible: at this level of linkability, the inability to detect Sybil attacks makes it extremely difficult, if not impossible, to build a structured detection mechanism. Unfortunately, this means that there is a necessary trade-off between privacy as opposed to security and functionality – necessarily, we are forced to sacrifice theoretically perfect privacy in order to gain functionality and security. However, we remark that even in this scenario, tracking is possible by simply following a vehicle. Because tracking vehicles by following them individually is unavoidable, it serves as a good baseline against which privacy can be tested. Detection schemes that require no linkability will definitely not negatively impact the privacy of the system.

2.3.4 Detection Approaches

Within the taxonomy, several sub-branches can be identified that relate to the specifics of each detector, such as what data the decisions are based on. The sub-branches discussed here are developed based on the literature listed in Tables 2.1 and 2.2; a detailed discussion of each individual scheme is provided in our survey article [142, 170].

Data-centric detection: Plausibility

Plausibility checks can be used to quickly and efficiently filter packets that are malicious. Typically simple instances of these mechanisms are assumed to exist by node-centric schemes in order to provide a way to determine trustworthiness of nodes. However, plausibility checks can also be used as a more advanced tool to determine a numeric plausibility value, rather than just filtering out bad packets. For example, one can analyze the speed or location of a vehicle over time, a receiver can identify its path and attempt to identify suspicious paths. Plausibility checks are often used to detect attacks that involve Sybil nodes, as such situations still require that the attacker transmits from approximately the same location, despite the usage of many different identities.

Signal-based plausibility One of the first applications for data-centric misbehavior detection was the identification of falsified positions by exploiting channel properties. By detecting the source direction of the signal and using time-of-flight, vehicles can attempt to verify the position contained within a CAM. An overall challenge of these approaches is that the error of measurements outside a controlled environment are very high [20], which is why these schemes are often combined with other approaches.

Multi-rule plausibility Another early idea was to use several sources of information to verify message validity, rather than just using signals to verify a position, which was sketched in [13] by Leinmüller et al., who later proposed a framework named VEBAS (see Section 2.4.1). The authors introduce the concept to use on-board vehicle sensor data and data from different layers of the communication system to build a world model, with which newly received information can be compared. Many of these multi-rule mechanisms also include some form of other detection mechanism, but often the focus lies on taking advantage of the many fast and simple checks that can be performed through plausibility checks.

Position Prediction As discussed in Section 2.3.4, the advantages of node-based mechanisms are strengthened when pseudonyms can be resolved, or at least linked, in the local vicinity of a vehicle. Position prediction essentially uses such techniques to predict behavior of vehicles and verify whether they follow an expected pattern. Although the Kalman filter is the most common approach, alternatives include the use of particle filters, as discussed by Bißmeyer et al. [88], and zone prediction, as discussed by Barnwal et al. [87].

Post-event detection The idea of post-event detection is that events are not isolated from each other, but typically relate directly to future behavior. A good example of such a scheme is the work by Ghosh et al. [49], who looked at the post-crash notification (PCN) application. After sending a PCN message, drivers normally adapt their behavior to avoid the crash site; this is exploited here to confirm whether the event was valid or not. Although this specific scheme uniquely uses driver behavior for validation, other proposals have been made that use subsequent beacons to verify whether an event actually happened (e.g., Ruj et al. [79] above). In their scheme, Ghosh et al. [49] use a technique called root cause analysis to detect which part of the event message was false. Although lacking in privacy, this scheme is very useful to serve as a baseline for revocation. However, it may struggle if sufficiently complex models of driver behavior are not available, and it notably cannot prevent false information from reaching the driver, which may lead to low user acceptance.

Data-centric Detection: Consistency

Consistency-based mechanisms look at packets or message sequences originating from distinct vehicles. These mechanisms focus on detecting and resolving conflicting information to achieve an accurate representation of the real world scenario. They are often employed by secure aggregation mechanisms to

Table 2.1: Data-centric detection approaches. The two categories marked *Data Trust-based* determine the trustworthiness of *events* based on both node- and data-centric trust; the same techniques could be used to establish trust in nodes.

Detection Technique	Category	Examples
Signal-based plausibility	Plausibility	Guette & Ducourthial [21], Ruj et al. [79], Xiao et al. [20]
Multi-rule plausibility	Plausibility	Leinmüller et al. [30], PVN [23], VEBAS [35], Yan et al. [37]
Position prediction	Plausibility	Jaeger et al. [73, 82], Bißmeyer et al. [88], Barnwal et al. [87]
Post-event validation	Plausibility	Ghosh et al. [49]
Raw data comparison	Consistency	Bißmeyer et al. [45], Zaidi et al. [149], Grover et al. [70], Leinmüller et al. [30]
World modeling	Consistency	Golle et al. [12], Dietzel et al. [46], Bißmeyer et al. [89]
Rule mining	Consistency	VARM [77]
Machine learning	Consistency	Grover et al. [71]
Sensor sharing detection	Consistency	Yan et al. [85]
Decision logics	Data Trust-based	Raya et al. [33], Rawat et al. [76]

combine information from several vehicles into aggregates and to deal with inaccuracies, which may occur when aggregation mechanisms are used.

Raw data comparison The simplest type of consistency mechanism is direct comparison between message contents to check them for conflicts. This is closely related to consensus mechanisms from the node-centric mechanisms section: the mechanisms here are distinguished by the fact that a decision is mainly based on the data, rather than trust (potentially derived from data) in vehicles. Unlike consensus mechanisms, direct checking mechanisms often simply output that a conflict exists (although some authors add some trust mechanism to verify the effectiveness of their scheme directly).

World modeling Rather than comparing raw data, some authors suggest the more abstract approach of world modeling. In world modeling, the idea is to store all received data and keep all the conflicting information in a database. This removes the restriction of comparison between specific messages, and rather allows queries such as "is this world state consistent?" or "is this hypothesis true"? For detection, these approaches often aim to find the data that is the cause of a conflict in the consistency of the model.

Rule mining Finally, some authors have proposed mechanisms that use techniques from the field of data mining to extract useful information from a large amount of stored data, such as CAMs and DENMs. By extracting these rules, the likelihood of the message being valid can be expressed.

Machine learning One other area that has gained recent interest throughout security is the application of tools from the area of machine learning. Although we feel that machine learning will likely have its place in our field in the long term, we remark that the blind application of machine learning techniques to detect attacks is not a reliable one. Recent work on adversarial machine learning has shown that the current situation is volatile [161], and older work from the area of intrusion has already indicated that real-world performance of machine learned detection has not lived up to expectations [60]. As vehicular networks are still a class of networks that is under development, there is a significant lack of reliable available datasets [172]. Since the detection performance of machine learning models is typically bounded by the input data quality and availability, this suggests over-fitting might be a significant issue.

Decision logic To establish the trustworthiness of vehicles, which is used as input by the reputation mechanisms discussed above, authors often use a decision logic. These mechanisms are on the border between node-centric and data-centric, as they use techniques from trust management to determine the trustworthiness of *events*. The techniques of decision logics are different from most other data-centric mechanisms in that they are much more generic and not bound to a specific data type, and the combination with node-centric evidence is much easier.

Node-centric detection: Behavioral

Behavioral mechanisms are focused on the behavior of a particular node. This mainly concerns packet headers and meta-information like message frequency. Behavioral schemes in C-ITSs typically focus on identifying nodes which send messages too frequently or nodes which modify the message content in

Table 2.2: Node-centric detection approaches.

Detection Technique	Category	Examples
Watchdogs	Behavioral	Hortelano et al. [51]
Flooding detection	Behavioral	Hamieh et al. [39], Puñal et al. [93]
Central Sybil detection	Trust-based	Chen et al. [38], P^{2}DAP [86], and Footprint [90]
Node validation and reputation	Trust-based	LEAVE [25], OREN [64], SLEP/PRP [43]
Event validation and consensus	Trust-based	PoR [27], z-smallest [72], Leinmüller et al. [56], CoE [53], Petit et al. [75]

a way that does not adhere to protocol standards. As these attacks are not fundamentally different from attacks that some classes of network intrusion detection mechanisms aim at, there are not many C-ITS-specific schemes available. Behavioral mechanisms are especially popular to protect networks where routing attacks and fairness play an important role, such as MANETs; some of these misbehavior detection mechanisms have been adapted to work in C-ITSs scenarios.

Watchdogs Watchdogs are the primary example of behavioral misbehavior detection mechanisms, as discussed in Section 2.3.1. There are many proposals that adapt the original Watchdog mechanism for specific requirements of C-ITSs, because they have the advantage that they are fully independent of transported content. However, they can clearly only be applied in settings where multi-hop communication occurs (either through routing or flooding-style approaches). One such work is that of Hortelano et al. [51], who set out to evaluate the usefulness of the traditional Watchdog mechanism for C-ITSs. The core assumption is that routing is standardized, and vehicles can predict the (expected) behavior of their neighbors. To accommodate packet collisions and noise on the wireless medium, the required number of re-broadcasts is lowered by a certain threshold (called *tolerance threshold*), to reduce the number of false positives, similar to the MANET case. In addition, the importance of older results degrades more quickly over time, to account for the increased mobility (the authors call this *devaluation*). Once the Watchdog detects malicious behavior, it is logged in a local file, but reports are not forwarded to other vehicles or a centralized instance. Their evaluation shows that it is difficult to find a global threshold for deciding at which point misbehavior should be detected; setting this threshold dynamically could lead to a higher accuracy. In addition, we note this paper does not address privacy, making the true suitability for C-ITSs unclear. As pointed out by the authors, several vulnerabilities remain; additionally, there is no protection against Sybil attacks. We note that in our more general attacker model, the attacker could use selective jamming in addition to further amplify several attacks, which would further reduce the performance of this scheme.

Flooding detection In addition to mechanisms that are designed to detect attacks on routing, there are mechanisms that detect a related class of attacks: flooding and jamming. Although Watchdogs are capable of detecting some types of flooding, we create a separate category for detection algorithms that detect other types of flooding, such as jamming and MAC attacks. In contrast to Watchdogs, Hamieh et al. [39] describe a detection mechanism for jamming attacks specifically, based on detecting patterns in radio interference. They focus on a particularly difficult type of jamming attack, where the attacker behaves intelligently and jams only specific messages, rather than constantly jamming the entire channel. The attacker listens to the channel and jams specific messages as they are seen on the channel, which is known as selective jamming and is significantly harder to detect. Hamieh et al. propose that a correlation coefficient between correct reception time and time where reception errors occur can be used to determine a degree of jamming. However, since reception errors can occur randomly, the medium can only be considered jammed if the correlation is unusually high. A deeper discussion and classification of jamming attacks is provided in a later work by Puñal et al. [93], which discusses different types of jamming attacks and analyzes their feasibility. In particular, they show that some types of jammers are capable enough to prevent communication altogether with high probability, illustrating the necessity of jamming detection.

Node-centric detection: Trust-based

Similar to behavioral mechanisms, many trust-based mechanisms for C-ITSs are rooted in mechanisms that were developed for MANETs. These partially evolved from mechanisms such as the Watchdog [8] (discussed in the previous section), which provide metrics to establish the trustworthiness of a node.

To aggregate this trust, distribute it among nodes, and provide it to a back-end system, a mechanism is required that not only filters malicious nodes as quickly and efficiently as possible, but also prevent attacks on the mechanism itself. For example, Pathrater [8] aggregates the Watchdog results, but it may be attacked through Sybil attacks (as the authors also discuss). Core issues for trust-based mechanisms are Sybil attacks on the one hand, and high mobility and brief connectivity on the other. These challenges are much stronger in C-ITSs, as the connectivity between vehicles is sporadic, and privacy requirements lead to a vehicle being allowed to use multiple identities. This is especially significant when conflicting information is received from two sources that are either equally trustworthy, or both unknown up to this point. To account for these challenges, authors often propose the use of data-centric mechanism output, which they then integrate into the trust-based mechanism; we will discuss these approaches towards the end of this section. We include these mechanisms here, because the primary focus is the development of the trust management, and not the data-centric detection process. There are also dedicated surveys for trust schemes to be found in the literature, such as the work by Rivas et al. [78].

RSU-supported Sybil attack detection There are many schemes that operate using a centralized authority to detect Sybil nodes. The approaches we describe here are based primarily on interactions between infrastructure and vehicle; by grouping vehicles based on the RSUs they interact with, the assumption is that the Sybils of an attacker will always follow the same interaction path as the attacker.

Node validation and Reputation Following are multiple papers specifically aiming at mechanisms that use explicit voting. Here, voting is used to decide which nodes are (not) trustworthy. These are distinct from consensus schemes, which use identities to vote on whether or not a claimed event is true; such schemes are discussed below. Voting mechanisms strictly require protection against Sybil attacks, and those attacks are thus usually not part of the attacker model.

Event validation and consensus A common challenge in trust-based mechanisms is the application of trust over multiple hops. Event validation addresses this question by allowing each vehicle to use their identity as a vote in support of or against a specific event. Similarly, consensus mechanisms do the same, but often relate more closely to data-centric mechanisms (e.g., agreeing on specific data, rather than events, or including data-centric detection results).

2.3.5 Early Framework Proposals

Having discussed different directions for detection, we now briefly revisit earlier ideas to create frameworks for misbehavior detection, before discussing more complete proposals in detail.

Detection and Correction of Malicious Data

Golle et al. [12] have presented the earliest example of data-centric detection, which checks for consistency between assertions (messages, also called observed events in the paper) using some abstract model of the C-ITS. If this results in an invalid state, their system identifies possible explanations where a subset of all assertions is valid – the best explanation can then be chosen. The model relies on four core assumptions: nodes can bind observations to received communication, they can uniquely identify neighboring vehicles (that is, detect Sybil attacks using physical properties), they can authenticate to one another, and finally, the network graph should always be connected. In case inconsistencies are found, Occam's Razor is applied, meaning that the explanation with the least amount of attackers best explains the conflicts found. As an example of how their model works, the authors introduce two models that determine the correct locations of vehicles in the network. Although this is one of the earliest works on data-centric security mechanisms in C-ITSs, the outline of their mechanism is still used as a guideline for data-centric security and secure aggregation schemes. Unfortunately, the detection component of this work is not evaluated for feasibility, due to an assumption that neighbors can immediately exchange derived information. As the original authors note, assuming this kind of connectivity is not very realistic, as the amount of data exchanged in their scheme is quite high and available bandwidth in C-ITSs is limited. Other challenges in actually deploying a system such as this include the fact that some meaningful hypothesis generation needs to be implemented, and the potentially massive computational requirements. Finally, the model over-simplifies some of the world: for example, it is possible that attackers generate

messages that are consistent with the model of each receiving vehicle, but that still achieve the goal of the attacker. This is because, for example in terms of false positions, a lot of malicious observations do not necessarily conflict with any other observations. When combined with a Sybil attack, the decision based on Occam's Razor may even lead to incorrectly labeling legitimate vehicles as malicious. This is particularly problematic in low-density settings.

Plausibility Validation Network

Lo & Tsai [23] present a plausibility validation network, which aims to detect a type of data injection attack called the illusion attack, where the attacker injects false information into the C-ITSs. To protect against this attack, the authors propose a plausibility validation network (PVN), which consists mainly of a checking module and a rule database. The rule database contains a set of rules that govern whether certain information should be considered valid or not, by analyzing the individual message fields. The authors provide a list of such rules to detect fake vehicles, which includes dropping of duplicate messages, the position being in transmission range, valid time stamps and others. The set of rules is different per message type, and is default-deny (i.e., all rules must accept the message). In some ways, this is similar to later approaches taken by frameworks discussed in the next section; however, the type of rules specified is quite different. For example, PVN's rules are much more focused on things such as duplicate messages and hop count; the plausibilities of location, time stamp and velocity are relatively loose. Location plausibility is defined primarily through hop count (specifically, the distance traveled by the message must be further than the distance between the event and its sender). In many ways, the idea is thus similar to that of VEBAS; use a broad scope of detection mechanisms to detect every attack in isolation, rather than trying to build a single system that can detect any deviation. Unfortunately, this paper does not provide any evaluation of these rules.

2.4 Related Frameworks

2.4.1 Vehicle Behavior Analysis (VEBAS)

The work by Leinmüller et al & Schmidt et al [17, 30, 35] presents a wide collection of primarily data-centric position verification mechanisms. In their work, the authors differentiate between different classes of verification mechanisms, including autonomous mechanisms (referred to as *plausibility* in this work), cooperative mechanisms (referred to as *consistency* in this work), behavioral mechanisms, and a simple fusion and trust evaluation scheme. Their system evaluates incoming messages sequentially, accepting or discarding messages as they come in. The authors later also split their mechanisms in *positive* and *negative* checks, where the decision states that the message should be accepted or rejected, respectively. The authors use an exponential weighted averaging to combine the individual outputs of each mechanism (which is numerical and either in $[0, 1]$ or $[-1, 0]$ depending on its type). The resulting local aggregate is considered a trust value in VEBAS, which is periodically broadcast to one's neighbors as the local trust in the associated vehicle. This can then be fused with all trust perceptions received from other vehicles to determine the trustworthiness of vehicles. However, the authors point out that the raw fused output should be used to make decisions on messages, and not the fused trust (i.e., local aggregate). This is because the latter may be manipulated by other vehicles, potentially making the system vulnerable to a variety of reputation attacks.

The authors' work includes the following behavioral, plausibility and consistency mechanisms as input for their trust of neighboring vehicles: Acceptance range threshold (ART), mobility grade threshold (MGT), maximum density threshold (MDT), minimum distance moved (MDM), map-based plausibility, position claim overhearing, maximum beaconing frequency (MBF), pro-active neighbor exchange, as well as positive and negative movement validation. ART provides a very simple check that discards messages with position claims that are far outside the communication range of the receiver, and thus likely contain a falsified position. MGT checks the distance moved between two beacons for suspiciously high speeds. MDT examines the maximum amount of vehicles in a particular area; when too many beacons are sent from one area, the vehicle ignores further beacons from the same area, in order to limit the impact of Sybil attacks. MDM checks whether a vehicle has moved a minimum distance in a certain time frame, which is designed to detect road-side attackers. Map-based verification assigns a plausibility value to the received beacons by comparing the location to a road map. Position claim overhearing is specific

to (geo)routing scenarios: by comparing different overheard packets and their destinations, overheard packets can provide indications of a false position in the past. MBF: if a vehicle sends messages with an unusually high frequency, the behavior can be regarded as a possible attack. Pro-active neighbor exchange assumes vehicles periodically broadcast their 1-hop neighbor information, which can then be used to validate messages in more detail. Positive and negative movement validation respectively check whether the movement observed from a vehicle is realistic (positive) or abnormal (negative).

The authors combine the outputs of a detection mechanism over time by using the exponentially weighted moving average (EWMA) method. Here, older information is integrated with less weight than fresh information, and different weight can be given to different modules. How time is made discrete is not elaborated upon; the most reasonable assumption is that the expected message interval (i.e., defined by the beaconing frequency) is used to represent one interval over which a decay is made. The resulting output is fused through simple addition: this is possible due to the way that the different modules are attuned to one-another. The fused result decides the legitimacy of the message, and it can be used to generate trust. This includes the possibility to send and receive reports from other nodes in the network (which adds a trust-based element to the mechanism): once a vehicle has accumulated a defined amount of information about surrounding node behavior, it will broadcast the results within the 1-hop neighborhood. However, these recommendation reports are not trusted blindly by receiving nodes to prevent attackers from broadcasting fake positive behavior reports.

VEBAS provides detection for relatively specific attacker classes, such as a road-side attacker, which is a malicious entity that uses stolen credentials to its benefit (e.g., attacking message routing, or other position-related attacks). Although the author's results appear to be very good, we believe there is room for significant improvement in the strength of the attacker model, as well as the overall evaluation. In addition, this approach could be extended significantly to include detection of other types of attacks, such as those on aggregated data. One significant benefit of this work is that the proposed idea can behave in a privacy-preserving manner, as long as locally related messages can be temporarily linked (i.e., Mix Zones [134] can be overcome locally).

2.4.2 Multilayered security and privacy protection in Car-to-X networks

In their thesis [108], Hagen Stübing proposed a framework for mobility data verification in the local neighborhood of the vehicle. The primary goal of this work is to support local tracking of nearby vehicles, which is important for safety applications. To this end, the authors propose a framework designed around a well-studied concept from the field of engineering: the Kalman filter. This filter was originally developed for missile tracking [10] and has since found applications in GPS location estimation. The basic idea is that a linear state estimation is performed on location and speed information, whose error converges to zero under linear acceleration behavior of the tracked object (i.e., in this setting, a vehicle). This is done by assuming normally distributed noise (around 0) for both sensing and state transitions. The core idea in Stübing's work is that the Kalman filter can be used to estimate real states, and therefore detect attacker states and erroneous states by comparing estimated states to the next received state (i.e., the next beacon message). By applying additional checks, one can filter out beacon messages that cause the estimated state to deviate from the real state. On the other hand, vehicles may at some point change their pseudonym, causing a receiver to perceive them as a new vehicle: this is resolved through the use of the Kalman filter as well. To further improve accuracy, it has also been suggested that a maneuver recognition should be integrated into the framework, which allows for the detection of highly non-linear movements (e.g., breaking, lane changes or sharp turns). An overview of their framework can be found in Figure 2.3.

Stübing et al. [61, 73] verify transmitted CAMs by analyzing the sequence of messages to find the trajectory of each vehicle. By tracking a vehicle using a Kalman filter, they can verify the location contained within each CAM, thereby allowing the detection and correction of falsified data in CAMs. This works, because Kalman filters allow the accurate prediction of movement even under the influence of errors (e.g., they are also used to correct errors in GPS measurements). As a result, a Kalman filter allows vehicles to locally link pseudonyms with high probability, and features adjustment for errors and new vehicles. By defeating pseudonyms in this way, vehicles can check that vehicles are transmitting valid messages. This scheme explicitly does not distinguish between malicious and faulty nodes, instead aiming to detect any misbehavior. We note that the existence of Kalman filters does not imply that privacy is void – the Kalman filter only provides accurate estimates when actually following a vehicle

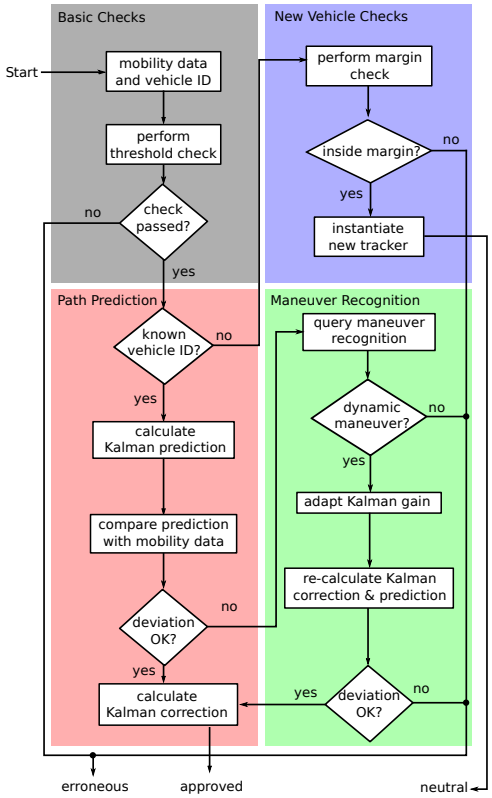


Figure 2.3: Stübing's framework, as illustrated in [82]

(similar to physically following it by driving behind it). The scheme was later extended [82] to deal with situations where Kalman filter accuracy is poor, such as lane changes and other special maneuvers. After computing the plausibility of a CAM according to the Kalman filter associated with the sender, they propose a scheme that probabilistically recognizes the maneuvers of a vehicle using hidden Markov models. A hidden Markov model is trained for each maneuver, classified by the Baum-Welch algorithm, allowing the state of the hidden Markov model to represent the different steps in the maneuver. The information used for these models is then used to update the Kalman filters where necessary.

2.4.3 Data-Centric Trust in Ephemeral Networks

In their thesis [40], Maxim Raya looked at a variety of different aspects of ephemeral networks, which VANETs as one of the prime examples. Unlike previous work, this thesis is much more focused on trust aspects: in the proposed framework, decisions are generally made based on node information. Therefore, a significant part of the work is a series of contributions related to trust networks, analyzing these with game-theoretic models. The one part of the work that analyzes data-centric trust establishment [33], which uses both data- and node-centric elements to determine the trustworthiness of nodes.

Raya et al. [25] have also been among the first to present a system for locally evicting nodes, including the possibility to perform global revocation as a result using a protocol called *LEAVE*. Here, vehicles exchange accusations about potential attackers; as soon as a threshold is reached, a vehicle is evicted temporarily. The set of accusations can be distributed in an aggregated message, which can be used to immediately ignore a vehicle, and used as a tool for global revocation once it reaches the certificate authority (CA). A core advantage of this approach compared to reputation systems is reduced detection latency, because trust does not need to be built over time in LEAVE. The authors argued that local eviction is especially suitable for vehicular networks because of the low communication overhead and quick reaction time to attacks compared to global revocation. However, global revocation based on analysis of the collected local reports is foreseen as an orthogonal countermeasure against persistent attackers. Some disadvantages of the scheme is that it may be vulnerable to Sybil attacks and may have privacy issues, depending on the type and implementation of the pseudonyms that are used.

Raya et al. [58] and Bilogrevic et al. [44, 64] have presented closely related game-theoretic approaches to combine their eviction scheme from [25] with a suicide mechanism similar to that of Zhuo et al. [43], combining the previous approaches. Here, a suicide action results in the immediate revocation of both accuser and accused by all receivers of this action. Apart from this action, vehicles have the alternative to vote by transmitting an accusation, or abstain from the protocol completely. Each of these actions is associated with a certain cost, where suicide has the highest cost. In addition, cost values take into account the fact that abstaining from any action for longer periods of time will leave the attacker more time to do harm to the network. Using game theory, each vehicle decides when and whether to take which action in order to minimize the cost for all participants using a variety of strategies.

Criticizing a number of existing revocation schemes including the aforementioned papers, Liu et al.'s [57] goal is to identify the limits of such schemes in vehicular networks. One of the main points argued in the paper is that a local honest majority, an assumption made by the revocation schemes discussed so far, is not as likely as the previous authors argued. If an attacker manages to create a local majority, they can create false accusations and falsely remove honest vehicles from the local communication network. If pseudonyms are used to protect user privacy, the authors considered it feasible for an attacker to use multiple pseudonyms in parallel to create such a local majority. A general conclusion for all voting-based schemes – both for revocation and other decisions – is thus that the parallel use of multiple pseudonyms should be prevented by the underlying pseudonym mechanism used. Moreover, Liu criticized Raya et al.'s game-theoretic approach discussed above, arguing that the assumption that each vehicle acts in it's own interest might be flawed, because the software running on the vehicles is not programmed by the vehicle owner themselves, but by its manufacturer. The authors note that manufacturers may optimize their own cost functions, which could invalidate the game-theoretic results that were obtained by assuming that all vehicles optimize their individual costs.

To establish the trustworthiness of vehicles, which is used as input by the reputation mechanisms discussed above, authors often use a decision logic. This is similar to traditional trust network approaches, which are typically adopted for this purpose. Raya et al. [33] present one such adaptation, which uses both data-centric and node-centric information to establish trust in specific nodes. However, the final decision is made by deciding on the overall trustworthiness of a node, rather than deciding the individual trustworthiness of messages. The authors used a combination of three different factors to evaluate trust in nodes: default trustworthiness, which is based on the type of certificate (e.g., police cars); event- or task-specific trustworthiness, which matches the type of vehicle to the event; and dynamic trustworthiness, which captures message-specific information like proximity to the event. Once each factor is computed, it is combined with the data-centric information and only then does the decision logic decide whether or not to trust the given node. The authors evaluated a number of different decision logic implementations, but stated that no single mechanism performs best in all simulated configurations. However, the Dempster-Shafer inference [1, 2] was identified as the most promising technique. This aspect will be discussed in more detail in the fusion and trust chapters of the thesis. Some of these authors have continued this line of work, and applied DST to the domain of participatory sensing [131].

2.4.4 Misbehavior detection and attacker identification in vehicular ad-hoc networks

Norbert Bißmeyer, in their PhD thesis [111] also studied the area of misbehavior detection. In their thesis, the focus was more broad than our thesis: it aimed at the detection of intrusions, as well as global revocation by identifying the attacker. To this end, the authors proposed a detection scheme for specific attacks (the overlapping position scheme), and extend this scheme in various ways to deal with position errors, as well as local vehicle tracking to improve performance. The scheme is also combined with attacker identification through trust mechanisms, which are then used for revocation. The revocation process uses an evidence-based approach [97], which served as an inspiration for the proposal made in our work to deal with reporting in a similar way [157].

In [45], the authors use consistency between CAM messages of vehicles to detect attacks. The authors claim that a model typically used for crash avoidance systems can also be used to determine likely attacks. The model describes that only one vehicle can occupy a given space at a given time. This allows detection of falsified position information. They assume it is possible to uniquely identify vehicles despite the use of pseudonyms (using the Kalman filter approach we discussed in the context of Stübing's work in Section 2.4.2) and use this capability to determine whether vehicles intersect. Intersection itself

is measured using the given width and length of a vehicle in a CAM. A mechanism to deal with errors is also introduced, by simulating larger rectangles outside the bounds of the vehicle. The size of each of these is computed using the speed of the vehicle. When detecting overlap of rectangles around different vehicles, a degree of certainty corresponding to the size of the rectangle is used. The higher this certainty, the higher the probability that the position for one of two vehicles was falsified. To compensate for lost packets, the authors also discuss a prediction mechanism to predict the location from which the next CAM is sent. To further increase detection accuracy, the authors note that a history of received CAMs and predictions can be used to compute a vehicles' trustworthiness; this trust value can then be used to decide which of two CAMs is the correct one. This scheme is highly efficient in terms of bandwidth, but the accuracy of the detection may be limited due to the effects of noise in the sensors. Although Sybil attacks are an issue for this work, the authors can deal with some degree of packet loss as well.

In their next work [89], the authors propose a more strongly centralized approach. They require each node to detect incidents, and forward them to a central authority that will detect misbehavior based on these reports. Because the central authority only needs to perform detection based on reports, rather than continuously for all nodes, this provides a scalable approach. In addition, this makes the assumption of an honest majority reasonable, when combined with revocation. Each node assigns trust to all nodes it knows; confidence in this trust is defined as a weight on the trust assigned by another node. For example, if node a has neighbors b, c and assigns a high trust to b, while b assigns a low trust to c, then confidence is used to describe the amount of evidence each node has for its trust assignment. These parameters are later used to determine reputation of each node by the central authority. The central authority can then resolve pseudonyms and determine whether any node was cheating repeatedly, or whether there was a benign fault. This is done by using the trust and confidence of the suspect and reported nodes; attackers are detected by determining which nodes have very low reputation.

The work continues [88], with a proposal to use a particle filter to represent the trust in a vehicle. This is done by using a particle filer that models all possible positions; the claimed position, as well as knowledge of the road and radar information are fused through addition. Afterwards, the normalization factor of the particle filter is mapped to a range of $[0, 1]$; how this mapping is performed is not specified, though the authors provide a number of suggestions. However, of primary note is that this information will all be based on some closed training phase, where "reasonable" values for the particle filter are learned. Trust is here modelled through an exponential weighted averaging ($\alpha = 0.5$) of message trust, which is the output of the above mapping process. In addition to this trust, a confidence is computed by determining the difference between the trust in a vehicle and the trust in the last received messages (from that vehicle). Confidence is used as a kind of threshold to determine whether or not the trust should be used or not.

Finally, the authors have discussed an experimental evaluation of misbehavior detection through CAM/DENM consistency [98]. However, this paper provides near-zero explanation on how the detection actually works, other than showing a single trace where a receiver detects position jumps, position overlaps and sudden vehicle appearances. The paper also explicitly mentions that a user is *not* incorrectly notified by a warning system, demonstrating that the author's goal was explicitly to warn human drivers. In our work, we also consider the setting where the vehicle's response is partially or fully automated.

2.5 Summary

This section summarizes the lessons learned from previous work, which is used to set up the design goals of Maat discussed in the next part of the thesis.

From VEBAS, we learn that many different plausibility schemes are likely to co-exist, and that combination of mechanisms is the only feasible way to detect the majority of real-world attacks. We take several of the plausibility schemes developed in this work, and extend or modify them to improve their performance. Through the application of subjective logic, inspired by the discussion of Raya[40] describing decision logics for trust establishment, we aim to thoroughly analyze the performance of such schemes. The question we try to answer here is, "to what extent can we rely on message-level fusion decisions, without the application of trust?", a question that, to our surprise, has remained unanswered by the related work discussed so far.

From Stübing [108], we take away the importance of working in a privacy-preserving way whenever possible, and the effectiveness of Kalman filters as a source for ideas. Although we do not explicitly

discuss privacy in our work, the selection process for detectors, as well as the literature survey above, aims for mechanisms that have weak linkability requirements. Our proposed system, Maat, will be designed to manage information about senders in a way closely analogous to Stübing's design, implicitly being able to use pseudonyms as identifiers. Kalman filters have been implemented as an example detection mechanism in Maat.

From Raya [40], our main take-aways include those from their work on decision logics, which pointed out that Dempster-Shafer theory (DST) is one of the most promising fusion approaches. In our work, we continue in this vein by applying subjective logic, a reasoning framework that encompasses DST. In addition, our trust setup is very much inspired by Raya's work, and results from LEAVE have influenced our design process significantly. For example, one of the reasons we decided an evidence-based reporting (leading to [157]) is useful, is the result that was presented by Raya [40]. Weaknesses of Raya's work [40] include a lack of details with regards to data-centric detection, the assumption that all attackers are rational, and a focus on long-term evaluation rather than focusing on the correct classification of messages. The latter case is much more suitable for C-ITSs due to the fact that these systems may respond to invalid messages before their revocation process successfully revokes a malicious actor.

From Bißmeyer [111], we learned the importance of central detection and the risk of introducing too many variables, as well as the value of high quality documentation and reproducibility of experiments. In their work, the authors introduced two misbehavior detection mechanisms, which due to the space constraints of the respective papers contain way too little details to be reproducible. Our failed attempts at reproducing their work show that this is likely due to the specific evaluation scenario chosen by these authors (although due to lack of data and code availability, it is impossible to validate this hypothesis). It should also be noted that because the authors have so many variables in their system, a rigorous evaluation of the effects of these variables is nearly impossible. In our work, we focus on correcting these mistakes, aiming to minimize the set of variables by isolating the different detection mechanisms for independent evaluation. However, the authors do also propose a number of interesting ideas, including one that we adopt: a central authority should be able to replay and reproduce detection results before trusting misbehavior reports. To this end, Maat's design is built in such a way that this is possible, by extracting a subset of our data structure that stores our detection results. These and many other issues will be addressed in detail in the next part, which describes the design of Maat, including the associated theoretical contributions.

Part II

Maat: Misbehavior Detection for C-ITS

Chapter 3

Overview

This chapter provides an overview of Maat, the framework that forms a core contribution of this thesis. The functionality offered by Maat is primarily centered around Research Question 2 and its sub-questions, as these concern data management and fusion. We first discuss some assumptions made in our work, and discuss some of the issues raised in the previous chapter, where we discussed earlier proposals for misbehavior detection frameworks. We also introduce the basics of subjective logic, the formal backing on which Maat's architecture is built, after which we discuss the architecture in detail. The remainder of this Part, consisting of four chapters, discusses the architecture components introduced in this chapter in detail, and contains the theoretical contributions of this thesis. The practical contributions of this thesis will be described in detail in the next Part, which discusses the evaluation methodology, and includes the various studies performed to evaluate Maat's performance.

Some sections of this chapter are based on our literature survey [170].

3.1 Background

This section explains the background that informs design decisions made during the development of Maat. These assumptions are based on the system model in Chapter 1, and in this section we primarily document how they relate to the components in Maat. We also provide a short introduction to subjective logic, which is extensively used in Maat; a more detailed introduction to the mathematical details is given as part of Chapter 5.

3.1.1 Assumptions

In this subsection, we discuss the assumptions made during the development of Maat. Although a lot of these topics have already been highlighted in the introductory part, it is useful to explicitly discuss the assumptions made by Maat. Maat was developed with current DSRC and ETSI G5 standards in mind, and has been inspired to a significant degree by work from the resilient aggregation domain [128]. We also look towards the future by considering more closely how misbehavior detection can work at different performance levels. We also assume a complete penetration of these technologies throughout the remainder of this work (i.e., all vehicles are equipped with some type of beaconing technology). However, Maat is designed to be technology-neutral: it is not fundamentally incompatible with recent developments in the area of 5G cellular communication, in which ideas have been proposed to have dedicated V2X communication bandwidth. This means that although the evaluation (part 3 of this thesis) is based on DSRC communication technology, adoption of Maat in a 5G-based VANET or a heterogeneous VANET (using both DSRC and cellular communication) would be possible. A more regular and high-throughput channel to central authorities may even significantly improve Maat's detection performance, since it is much easier to include this authority's recommendations in the decision process, and revocation is much more rapid (making it much harder to perform long-term attacks). However, we remark that in our opinion, the fundamental challenge in VANET research still lies in the de-central operation of as many system components as possible. A safety system should not primarily rely on the availability of networked services where it can be avoided, and similarly, the services for future C-ITSs should operate in a decentralized fashion where possible.

Although Maat assumes a decentralized network for communication, it is still assumed that the overall traffic system is subject to some regulation through the issuing of pseudonymous certificates that authorize a vehicle to transmit valid messages. This is in line with the various EU and US research projects and standardization activities that have been performed over the past decade, as discussed in Section 2.1.2. , which have focussed on how a sufficiently large set of pseudonyms can be distributed, how authentication can be performed efficiently, and how revocation can be performed with minimal contact with a central authority. The reason that Maat assumes this setting is that it prevents simple implementations of a variety of simplistic attacks (including impersonation and replay attacks). Although it may also be possible to use Maat to detect these attacks, it is much easier and more efficient to rely on well-documented cryptographic mechanisms. Maat should be considered as a layer on top of this cryptographic layer, which aims to add additional guarantees by detecting attacks that are performed with valid key material. As most currently proposed systems rely on single-hop broadcast messages and broadcast authentication mechanisms through ECDSA-signed messages with an attached pseudonymous certificate, Maat is primarily geared towards this scenario. Beaconing messages, as standardized in the SAE and ETSI standards, consist of few, well-documented fields whose contents can be verified independently (e.g., by measuring whether there is a vehicle at a claimed position). Maat can also be used to verify messages of other types, such as messages exchanged for in-network aggregation [103], but the primary focus is on security for applications built on top of a local dynamic map (LDM), which includes single- and multi-hop data. In our evaluation, we focus on CAM-based applications as the primary candidate, but our design transfers to any message type where the origin of the data is well-defined (including both types of BSMs and DENMs).

In order for Maat to function correctly, guaranteed message integrity and implicit linkability is required. This follows directly from the fact that Maat uses existing and newly developed detection mechanisms that validate messages by linking them to earlier messages from the same sender. If integrity is compromised, an attacker can easily cause chaos by falsifying messages sent by other actors in the network. Similarly, without at least some linkability between messages, the vast majority of detection mechanisms cannot work reliably; even many plausibility checks require at least two linked messages (see, e.g., VEBAS, as discussed in Section 2.4.1). Maat would otherwise be limited to a very small set of detection mechanisms that can validate messages independent of earlier messages. Although this is an interesting goal in itself, our focus is quite different. For the remainder of this work, we assume the linkability is explicit (i.e., through pseudonyms), which corresponds to the previously mentioned standards.

Maat could in principle be used to detect attacks on the network layer (e.g., blackhole and wormhole attacks), however these attacks are already well-understood [164]. Therefore, we focus on the layers above this one. Specifically, the main setting that Maat assumes are applications compatible with some limited latency. This enables a more complex detection process that can also be used as evidence for revocation, rather than just making safe decisions. The over-arching goal is to provide a service that manages reliable data on medium time scales (i.e., maximal delays of around 10-20 milliseconds), as opposed to (near) real-time applications that require faster response. Although Maat's techniques can also be applied in the latter setting, some of the advantages offered by Maat require detection mechanisms that use data gathered over time, which make their use in real-time applications challenging. For this thesis, we therefore concentrate on applications that are based on an LDM, as previously discussed in Chapter 1. These applications include those applying beacons in the long term, such as locally tracking nearby vehicles, as well as traffic information services and other long-term applications. An example of a real-time application for which Maat's detection is not designed, but for which its techniques are still beneficial, would be cooperative adaptive cruise control (CACC). In that application, the vehicle uses control algorithms to automatically adapt the vehicles' acceleration by exchanging messages with acceleration information: in such applications, several 10s of milliseconds of delay may lead to a significant performance impact.

Although this thesis focusses on the detection of malicious misbehavior, significant effort in the literature is also devoted to the detection of non-malicious errors. Throughout the work on this thesis, the distinction between malicious and erroneous misbehavior has been a topic of discussion [117], and also a topic that has received relatively little attention in the security research community. This is partly because the distinction between some errors and malicious messages is entirely philosophical: distinguishing these messages implies inferring intent of a sender. Presumably for this reason, most work does not attempt to distinguish between erroneous and malicious behavior, instead assuming that

all incorrect behavior is inherently malicious. We follow this line of reasoning and focus on detecting misbehavior, but without attributing malicious intent; we simply inform the application that accesses the data about the uncertainty due to misbehavior. This ensures that errors in the system are dealt with and significantly simplifies the design of Maat, since attribution of malicious intent is known to be difficult. An operator of a C-ITS may be interested in the distinction between errors and maliciousness: for this purpose, Maat aims to provide a chain of evidence for each detection it performs. Maat assumes that detectors are shared among entities in the system: as long as this assumption holds, it is possible to replay the chain of evidence and validate the conclusion reached by another instance of Maat. This should help humans make informed decisions when distinguishing errors from malicious misbehavior.

Finally, we briefly address the topic of jamming, intelligent or otherwise. Jamming refers to a type of lower layer denial of service attack (usually in the physical and/or link layers): by transmitting signals at the appropriate time, data exchange between specific vehicles can be disrupted, or in some cases even between all vehicles. In the physical layer case, transmitting a continuous signal can disrupt communications; in the link layer case, transmission during certain intervals is usually sufficient, by making the channel appear to be busy. Jamming is one of the most significant concerns for a real-world C-ITS in the context of CACC and other low-latency applications, but it is also difficult (if not impossible) to detect reliably. Although Maat can in principle detect missing messages that were not received, large scale jamming attacks that completely flood the network cannot be defended against using detection systems like Maat. For this thesis, jamming is therefore considered out of scope: however, we do deal with message loss (e.g., Maat will work if some fraction of messages is lost).

Returning to the literature systematization introduced in Chapter 2, we briefly address how Maat could be classified in this system. Maat allows, at least in principle, the integration of any type of misbehavior detection discussed in our taxonomy. However, in this thesis we primarily focus on data-centric, specifically plausibility mechanisms, which we consider to be the most useful. Their independence of other vehicles reduces the potential attack surface and simplifies independent evaluation, which are characteristics that were identified as useful in Chapter 2. In terms of detection scope, Maat is oriented towards local detection, but designed in such a way that results can easily be distributed between nodes and with a back-end given enough bandwidth. In our evaluation we primarily focus on Maat as a local detection component, but due to the reproducibility of the detection results, Maat is uniquely able to transfer the results effectively compared to existing work. In terms of privacy requirements, Maat was designed under the assumption that pseudonyms are valid for a reasonable period (on the order of minutes); we do not expect full anonymity (i.e., all messages are unlinkable). Designing mechanisms to link pseudonyms locally, as done by Stübing et al. (see Section 2.4.2), is conceptually foreseen for Maat (see Chapter 4), but was not implemented in our prototype.

3.1.2 Subjective Logic

This section provides an brief, high-level introduction to subjective logic, a formalism for reasoning under uncertainty. Subjective logic will serve as the foundation of reasoning in Maat; in the process of its development, we have also made contributions to subjective logic [171], which we will describe in more detail in Section 5.2. The development of subjective logic started in the late 1990s [6], where Audun Jøsang was developing a formalism for artificial reasoning and notions of trust. The logic is designed for trust management: it distinguishes between what we will refer to as variables, actors and opinions. A simplified example might be that two persons observe an event (e.g., looking out through a dirty window to see whether it is snowing) and convince a third person of their observation by offering some evidence. In this example, the opinion of a person represents their belief in a particular value of the variable X: person A may believe it is snowing ($X = \text{true}$), while person B may believe it is not snowing ($X = \text{false}$). If both try to convince person C, it seems natural to allow A and B to also express uncertainty about their opinion, and for C to express their trust in A and B as an opinion too. This is exactly what subjective logic aims to mathematically formalize and achieve.

For binary variables, opinions are described in many works as consisting of belief b, disbelief d, uncertainty u and a base rate a. We will discuss non-binary variables in Chapter 5. Formally, an actor A has an opinion over the outcome of a variable X that takes values from the domain $\mathbb{X} = \{\text{true}, \text{false}\}$ represented by a *binary opinion* $\omega_X^A = (b, d, u, a)$, with $b, d, u, a \in [0, 1]$ and $b + d + u = 1$. The base rate a represents the *a-priori* belief of A in the opinion when they have no evidence for one outcome over the other (i.e., $u = 1$). To compute the degree of belief from A in $X = \text{true}$, one can use the *expectation*,

which projects the space of opinions to a single value $e \in [0, 1]$, which can then be used as in probabilistic logic.

Subjective logic also provides operations for the combination of opinions, both in terms of reasoning (i.e., extensions of logical operators and new reasoning operators) and in terms of fusion and trust. Reasoning operators (e.g., inference and abduction operators) allow one to derive new conclusions about previously unknown facts, or to translate belief on one variable into belief on another, analogous to the way inference is used to update a knowledge database in artificial intelligence. In this work, we are mostly concerned with the fusion and trust operations, which allow actors to combine opinions about the same variable from multiple sources, and to maintain and update their trust in those sources. We thus focus our discussion on trust and fusion; an extensive overview of the five main operators, including definitions for the fusion of two opinion sources, can be found in the book on subjective logic [144]. In Chapter 9 we introduce generalizations into multi-source fusion operations for these operators and provide a much more detailed overview of subjective logic. For trust, subjective logic provides a discounting operator that can be used to translate trust in another actor into trust in information provided by that actor, which we apply in Maat to model the reliability of detectors.

The mechanics of fusion between different opinions depends significantly on information external to an opinion, such as whether two actors' observations are dependent (in the sense of probability theory), as well as the desired outcome. For this reason, multiple fusion operators are defined for this logic, and the appropriate choice should be made by the analyst [144, Chapter 12]. The list of available operators includes belief constraint fusion (BCF), cumulative belief fusion (CBF), averaging belief fusion (ABF), weighted belief fusion (WBF) and consensus & compromise fusion (CCF). With the development of Maat, it became necessary to provide multi-source extensions of these operators, which were not available for WBF and CCF. As part of the contribution of this thesis, we define these extensions; this will be discussed in detail in Chapter 5.

3.2 Maat

Maat is a misbehavior detection framework designed to combine the execution of multiple misbehavior detection mechanisms that can be designed independently. Unlike most previous work, it can conceptually integrate a large variety of different mechanisms, while maintaining flexibility over how the final result is computed. This is done through the application of *subjective logic*, a belief theory similar to Dempster-Shafer theory (DST). In this logic, *subjective opinions* are relations between objects or actors that express confidence or trust, along with a degree of uncertainty. Maat uses these opinions to express relations between data, other vehicles, and detection mechanisms. A natural data structure to store this information is a directed graph, since these relations are directional. Maat can then examine all paths that lead to a specific data item (e.g., a position claim of a vehicle) and draw conclusions from subjective logic through the fusion and transitivity operators provided by the logic. The graph is Maat's core; it is referred to as the *world model*, as it represents all the available knowledge of a specific vehicle that runs Maat.

Maat also provides components to extract the relevant data from messages, store the appropriate data in the graph, and manage the independent execution of detectors. These tasks are executed in separate threads and communicate through a graph-like structure through message passing. In this model, detection mechanisms execute independently: each sees the series of messages as a sequence of inputs, as it would when directly receiving messages. By expressing the output of these detection mechanisms as opinions on the data (or on other vehicles), Maat can store and combine the output of each mechanism. A versioning system is used within the graph to enable access to previously received information, as well as previously executed detection events. This is useful for generating messages that contain evidence, i.e., messages that capture *why* certain decisions were made. These messages can be used for reporting misbehavior to neighbors, or for global revocation by transmitting a subset of the graph to a central authority.

In Figure 3.1, a high-level overview of Maat is given. On top, the ad-hoc network between vehicles and road-side units is illustrated: each vehicle operates an on-board unit (OBU) with corresponding network stack. Within the network stack (which is based on the corresponding ETSI standard, ETSI EN 302 665), Maat is best placed at the facilities layer, because it serves as an interface between the applications and the underlying network and sensors. Maat's overall workflow is a single pipeline for

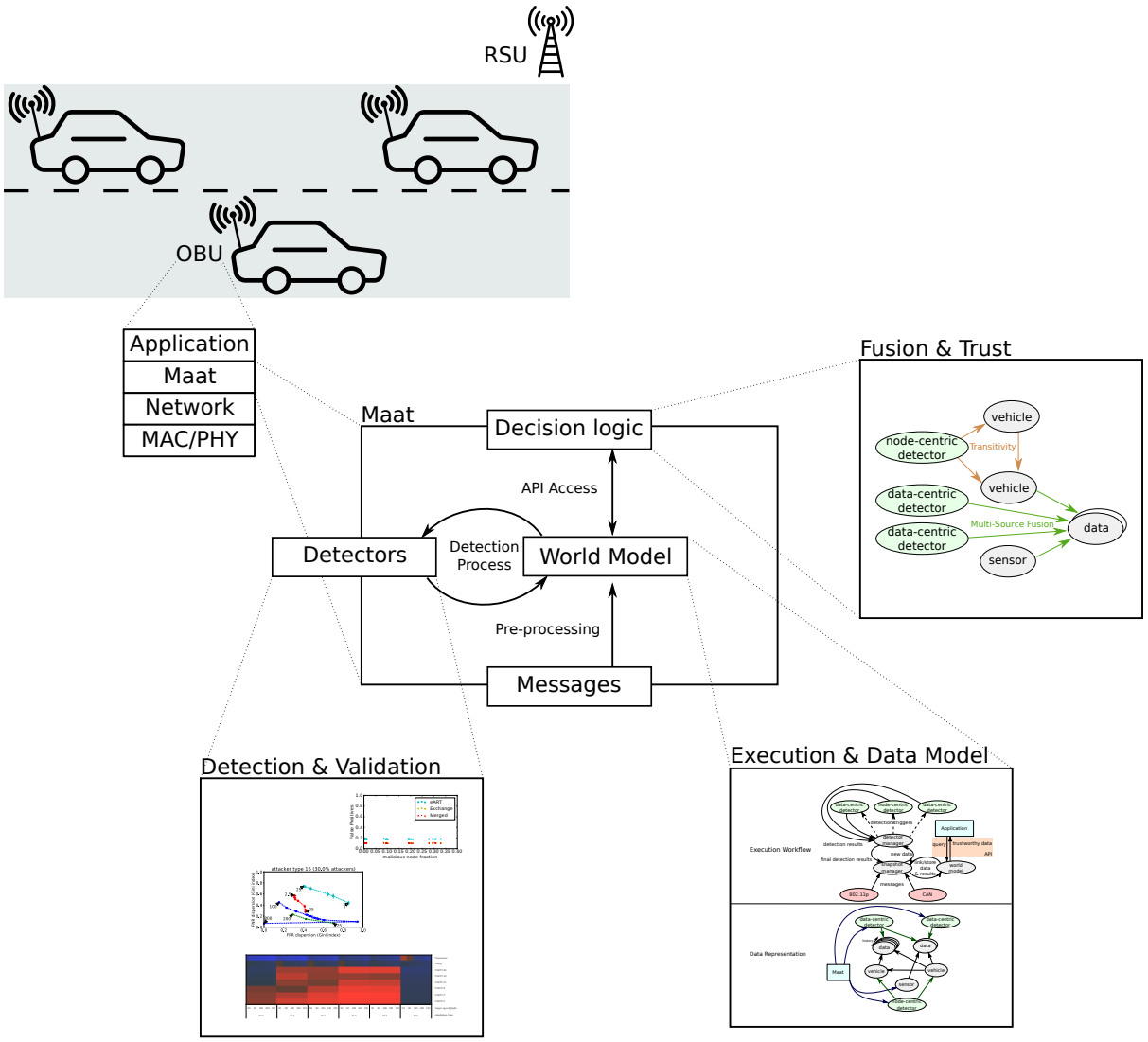


Figure 3.1: High-level overview of Maat

both CAN-internal and VANET/C-ITS communication, which provides a logical ordering for the entire system. Data is extracted from these messages using simple parsing code, creating or updating the world model as appropriate. The world model stores the data in a graph structure, including trust relations between vehicles and detection outcomes of the individual detectors. Maat ensures that all detectors work in parallel, in isolation of one-another, before storing all the results in the world model. Storage in Maat is versioned, i.e., it is possible to analyze specific historical states in the model. Maat provides an API, mediated by a decision logic, which provides applications with access to information exchanged within the network.

The detailed architecture of Maat has three components, also illustrated in Figure 3.1. The first component we will discuss is the bottom right component: the execution and data model. Conceptually, this component describes how data is stored and accessed by applications; in addition, this component is responsible for data management, locks, parallel execution and similar implementation details. We elaborate on these responsibilities in Section 3.2.1. The second component is responsible primarily for decision making: it controls what data is labelled trustworthy when an application asks for reliable data. This involves modeling the trust between vehicles, exchanged in the network, as well as the combination of detection results produced by detection mechanisms. Section 3.2.2 provides additional details on the questions this component needs to address. Finally, the detection & validation component contains the actual detection mechanisms developed for Maat. This component also controls what detectors are executed in which situation, and how much these detectors are relied on. This is described in Section 3.2.3. Each of these sections correspond to a detailed chapter in the Maat part; in the evaluation part, we discuss how these components interact to provide Maat's behavior.

3.2.1 Data Management

The data management component is centered around the world model, and basically describes how information is stored in Maat. As previously noted, a graph is the natural way to represent a world model that contains both data, entities that have opinions about these data, and those opinions themselves. We now discuss in some detail how the graph is structured; an example is given in Figure 3.2.

Data is stored in vertices of this directed graph that have no outgoing edges, labeled *data* in Figure 3.2. Opinions about the *correctness* of the data are the incoming edges of those vertices. Correctness can be claimed by a sender, e.g., a vehicle claims to be at a particular position: this is an edge between an entity and the data. Data vertices have a *referent*, which is the thing about which the data provides information (e.g., a region for a traffic jam notification, or a specific entity for a beacon message), in order to distinguish the referent from the sender. This allows consistent representation of forwarded messages. Other entities in the system can be other *vehicles* or *sensors*, also marked in grey in Figure 3.2 because they are also external information. Correctness of data or trustworthiness of entities are validated by detectors, i.e., detection mechanisms executed locally in this instance of Maat; these are data-centric detection mechanisms and node-centric detection mechanisms, respectively. These are marked in green on the top and bottom of Figure 3.2. If a trust scheme is used that explicitly exchanges trust between vehicles, this information is represented by edges between entities in Maat, such as the edge between the two grey vehicles in Figure 3.2. Finally, add a single root vertex that has no incoming edges, labeled Maat in Figure 3.2, issues opinions on detectors: these opinions can be used to model which detectors are currently active (i.e., whose opinions are actually considered when evaluating a query).

In summary, Maat thus contains four vertex types: data, entities, detectors and the root vertex (referred to as MDS, or misbehavior detection system, since it ultimately decides which misbehavior detection mechanisms are actually used). Edges relate the root to detectors, detectors to entities, detectors to data, entities to entities, and finally entities to data, for a total of five edge types. This can be seen in Figure 3.2, where two data items, two vehicles, a sensor, two data-centric detectors and one node-centric detector are shown. An example corresponding to this figure could be that the two data items represent the beacon information of the vehicles. In the next section, we discuss what happens when one of these beacon sets is accessed by an application through the decision logic. Whenever receiving a query for data, Maat automatically fuses the available opinions from all sources, as discussed in the section on fusion and trust.

However, before discussing these topics in detail, we first briefly discuss the execution model used in Maat, as shown in Figure 3.3. As already pointed out, the updating of the world model and the application API are independently executing components. Information may come from multiple sources,

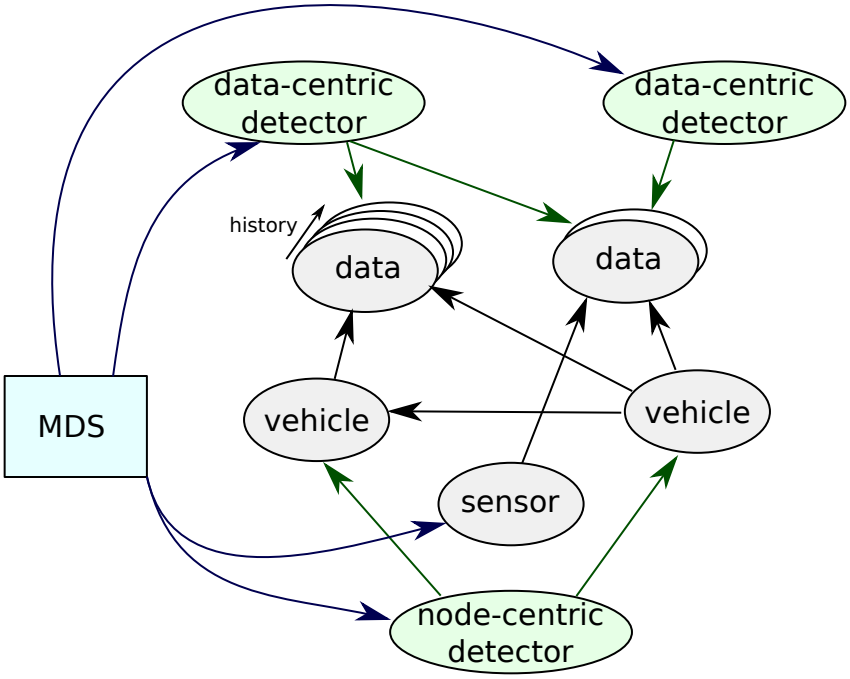


Figure 3.2: Maat's data model

arriving through the network stack: to have a consistent view, Maat uses a single entity to manage the arriving messages: the snapshot manager. This manager dispatches detection through the detector manager using a standard scatter/gather abstraction, querying each detector for their detection results. The detector results are collected and passed back to the snapshot manager, which then stores the data and results into the world model. In parallel, an application can access the world model, as shown to the right of the figure. A detailed discussion of data management and execution can be found in Chapter 4. We now move to discuss the API used to access the data, where the decision logic selects the data to be returned.

3.2.2 Fusion and Trust

Representing the received data and collecting opinions from detectors is not useful by itself: decisions on what data should be accepted and acted upon is an important step to actually build a working system. In Maat, this is implemented through two parallel approaches: allowing trustworthiness queries on any results that area already processed, and immediate message-level decisions as soon as detectors have made relevant decisions. The second approach is most useful for immediate intrusion detection, and most of this thesis will surround this approach; however, the fact that all relevant opinions are stored is a deliberate decision, made to open up future directions for research. By storing this data, detailed reports can be generated from Maat instances, and these reports can be completely verified and even re-generated by another instance of Maat. It also enables revision of previous output by detection mechanisms, which is a common issue in secure data aggregation[1].

Both the queries and the immediate decisions are implemented using a combination of fusion and transitive trust. Fusion is the combination of information from multiple sources: Maat uses fusion to combine evidence from different sources (i.e., incoming edges in the graph). Transitive trust, on the other hand, allows Maat to use trust received from other entities, and to interpret claims from other entities through node-centric detection mechanisms. Transitive trust is also used by Maat to weight the input from different detectors; this will be discussed in more detail in the next section. If Maat is deployed without node-centric detection mechanisms that explicitly exchange trust information, then there are no

[1]In summary: what happens if a legitimate vehicle transmits a message, which it later realizes, is invalid? In Maat's model, detection mechanisms can correctly handle these cases. Similarly, the literature has some proposals [79] that suggest vehicles also broadcast inverted claims if they learn their claims have become invalid: in Maat's model, detectors can revise their opinions about such claims.

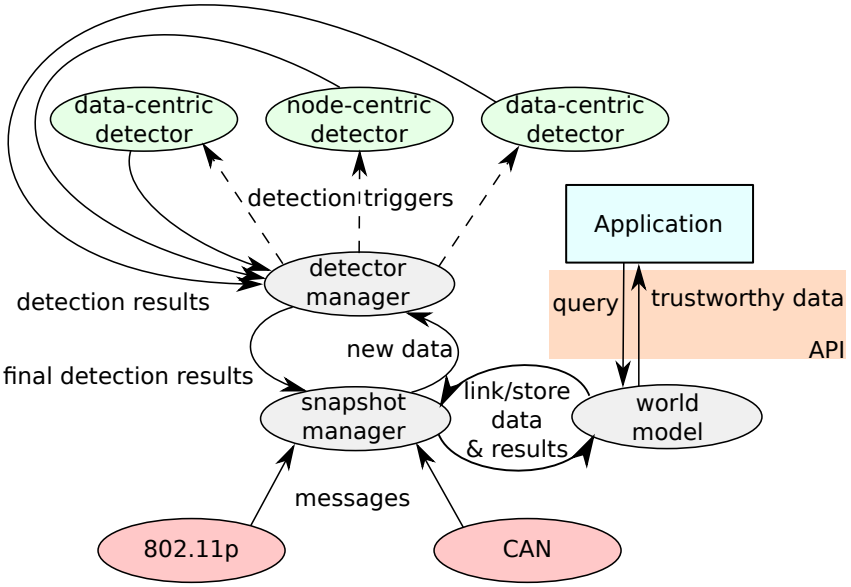


Figure 3.3: Maat's execution model

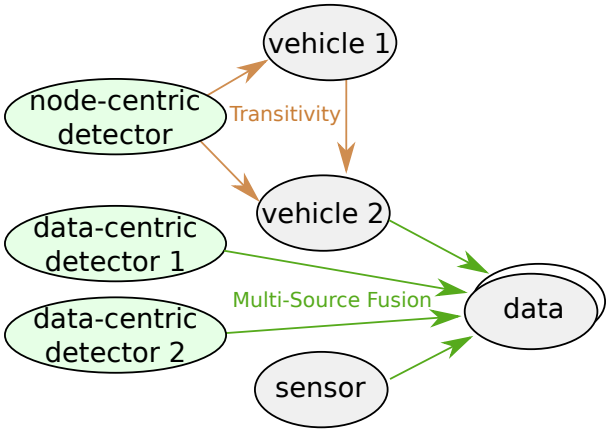


Figure 3.4: An example of fusion and trust transitivity.

edges between different entities, making the graph a directed acyclic graph. This implementation has advantages in terms of runtime performance, and avoids the thorny issue of cycles in a trust graph. Since arguments can be made that recommender trust over multiple hops is generally not very reliable, the information lost through this decision is quite limited.

For fusion, subjective logic offers multiple operations: binary operators have been proposed for these operations in the book on subjective logic in 2016 [144], as well as partial proposals in the literature [160]. However, not all of these operators are associative: part of the contribution of this thesis is the proposal of *multi-source fusion* operations[2], a contribution to data fusion that will be discussed in more detail in Chapter 5.

Figure 3.4 gives an example of what happens when the right-most data item in Figure 3.2 is queried: the returned opinions are listed here. The orange edges between the node-centric detectors and vehicles are used to compute the trustworthiness of the vehicle that has an opinion about the data. The resulting opinion from these trust operations is merged with opinions provided by data-centric detectors and sensors. This is done through fusion operators, as discussed previously. Note that the opinions between the root and the detectors are not included in this example; these are discussed in more detail in the next section, as they relate primarily to the detection process.

[2]We use *operation* to refer to the fusion of an arbitrary list of opinions from various sources, rather than *operator*, which we use to refer to n-ary operators for fixed n.

3.2.3 Detection

Accurate detection of attacks fundamentally depends on the performance of detectors used in the detection process. A well-known result in the field of data fusion states that no fusion operator can improve performance in all cases with imperfect detectors; this is referred to as *data imperfection* [105]. Therefore, this thesis also discusses several existing detection mechanisms and their extension to subjective logic for an optimal support of fusion. In the evaluation of Maat's components, we also provide a novel metric based on error rate distributions across observers, which is an indicator for the potential performance improvement that can be gained by improving the detector, rather than the fusion of different detectors. We explain how this metric, subjective logic, and detector design are closely related concepts that significantly impact overall performance in Chapter 6 and Chapter 8.

For the prototype of Maat, a number of different detectors were implemented and evaluated. Some of these were re-implemented from earlier work, as no public implementations exist for the vast majority of detection mechanisms in the literature[3]. It turns out that re-implementing and deploying detectors in the more elaborate evaluation scenarios that have arisen in the vehicular communication community since those mechanisms were proposed is quite valuable, as our results show several mechanisms to perform much more poorly than previously thought. Based on those results, we propose alterations to existing detectors, and we used this information to implement new detectors. We described these modifications and proposed detectors in Chapter 6.

One key fact about detectors appears to be that they are fundamentally bound to detect *specific* attacks, or rather validate specific information stored in the world model. Thus, the question arises how detectors can provide the fusion process with information on how reliable their decisions are, more specifically how the subjective opinions used for fusion are chosen. However, some detectors simply cannot perform well in certain situations, e.g., verifying whether the claimed speed is in a valid range for a given road will not be beneficial in a traffic jam on this road. It thus seems natural to avoid relying on the detectors alone to correctly estimate their confidence, since doing so assumes that all detectors correctly estimate confidence for every possible scenario. This is where trust transitivity on the detectors comes in, as mentioned in the previous section: trust transitivity can be used to externally manage the detectors, applying weights to the detectors for fusion. Chapter 6 will discuss in detail how trust transitivity is applied, and provides a theoretical perspective on trust transitivity and detector confidence, respectively. For the initial prototype of Maat, trust transitivity is used statically to show the interaction between detector performance, weights and fusion operations (see Chapter 8).

[3]To the best of our knowledge, ours are the only ones whose source code is publicly released

Chapter 4

Data Management

This chapter describes data management in Maat, as briefly introduced in the previous chapter. We describe the design of the world model, discuss the various decisions made during its design and describe how data is stored by the framework. This provides a foundation for the next chapter, which explains how data is retrieved from the world model, and how its trustworthiness is assessed.

4.1 World Model

This section addresses the conceptual design and motivation of Maat, discussing the underlying ideas followed by a formal description of our graphical representation.

4.1.1 Background

Before discussing misbehavior detection specifically, we discuss world models in the abstract. The idea to manage all related data for a vehicle at one point is relatively common for a variety of different applications and at different abstraction levels. For example, a network routing application typically maintains some kind of neighbor table [36], usually with some more information (e.g., position information). On the other hand, aggregation and secure aggregation frameworks [128] have been developed that take a much more holistic view, aggregating all dynamic information distributed in the network. A similar trend can be observed in standards; for example, ETSI standards for the transport & network layer [47, Section 6.2] and the facilities layer [47, Section 6.3] both include a data store that contains information about the vehicle's surroundings, for routing and service purposes respectively. Although the designs appear different at first glance, they have important commonalities: at both layers, information from the vehicle's surroundings are represented. In the aggregation literature, this is typically referred to as a *world model*, a model that explicitly stores and represents information about a vehicle's surroundings.

The concept of a world model is also used by works described in related work: both work on VEBAS (see Section 2.4.1) and the work by Raya et al. (see Section 2.4.3) includes a data store describing the vehicle's surroundings. Throughout the detection literature [12, 33], the concept of some central data structure is often applied implicitly, and this leads to considerable confusion at which point detection occurs. There are two different points at which decisions are typically made: immediately as data comes in, as part of message processing (e.g., in VEBAS), or at query time, when an application accesses the data (e.g., in secure aggregation work [128]). Depending on which of these is applied, the decision algorithm has more or less information available, which is important because the amount of available information represents the upper bound of possible performance.

When decisions are made as part of message processing, misbehavior detection can be seen as an application of *stream processing*. The use of stream processing for misbehavior detection often reflects ideas from intrusion detection, and is particularly useful for low latency use cases, where immediate decisions are required. With the emergence of cooperative autonomous driving, this approach is gaining ground, but due to its relative novelty, there are relatively few authors specifically addressing this topic. Stream processing can be further sub-divided into a filtering approach (as e.g., taken by Stübing et al. [82] with their plausibility checks) and a labeling approach (e.g., Watchdogs, which label specific nodes as suspicious). In a filtering approach, each assessment assumes that the world model only contains reliable

data, and this data is then used as a basis for the validation of new data. However, it has been suggested that classification errors in such approaches can cause the system to deviate from the real world, especially if the data is noisy [141]. These types of attacks are well-documented against aggregation scenarios in particular, as discussed in previous work [103, 128].

To counteract these attacks, there is other work that performs its decision making when data is accessed by an application [49, 75]. This means that whenever an application accesses data from the world model, the system determines whether the data is trustworthy. Depending on the specific application, it may then return only trustworthy data, label the data with a broader confidence range, or exclude it completely. Regardless of what is provided to the application, there are also multiple ways to evaluate the data. Previous work has looked at streaming information for applications, which is very close to stream processing as discussed above (e.g., Petit et al. [75]). Basically, applications here require streaming information, and the detection process directly influences how the application responds to potential safety-critical information.

On the other hand, both early and recent ideas in the literature suggest that hypothesis testing [12, 149] might be a more comprehensive approach. In these works, it is generally argued that more information implies better detection performance. Golle et al. [12] basically apply Occam's Razor, and select a subset of all messages that must be malicious as the minimum set so that the data in the world model is consistent. Zaidi et al. [149] propose a specific hypothesis testing method to compare the average traffic flow to the behavior of a specific vehicle; if it deviates too much, it is considered inconsistent (and thus malicious). Analogously, trust-related results from Raya et al. [40] are based on game theory, describing longer-term effects of potential attackers, and discussing the feasibility of long-term attacks in this style. All of these techniques rely on the fact that information is stored and kept as long as possible; this is one of the key inspirations for Maat's world model design, which assumes that data should be available permanently[1]. Similar ideas have previously been proposed based on a secure logging approach [110].

In Maat's detection, both stream processing and verification at data access are applied: as messages come in, each detector is started and provides its output to the world model. When the data is queried, all the relevant results are collected from the world model and combined together. Because Maat's detection process is completely parallel, as long as detectors obey the constraints outlined in this chapter, the detection results will be repeatable. It is also possible for Maat to request new detection results based on novel information, by injecting a corresponding message (e.g., to implement detectors such as the one proposed by Zaidi et al. [149]). Maat's overall workflow is shown in Figure 4.1, which is a substantial improvement of our initial idea [115]. This figure shows the interaction between the six components that make up Maat: an input converter that splits and converts the message into the appropriate format (1), allowing the snapshot manager to write the data to the world model and launch the detectors (2a), through the detector manager that supervises detector execution (2b). After the detectors finish their detection process, they return their results through the detector manager (3a) back to the snapshot manager (3b), which appends this data to the world model. The snapshot manager is responsible for ensuring that the world model is always in a consistent state, which is implemented through versioning as discussed in Section 4.2.2. A different view of this process was also described in the execution workflow in Section 3.2.1. The API on the bottom of Figure 4.1 represents an interface through which both Maat's components and applications have continuous read access to the data stored in the model.

In the remainder of this section, we discuss how the world model of Maat is structured and the reasoning behind it. Recall from Section 3.2.1 that the world model has a graph structure that connects the detection system itself with a number of different node- and data-centric detectors. Those detectors in turn evaluate other vehicles and the data they have transmitted, represented as entities and data items respectively. All of the resulting evaluations and trust relations are linked through opinions, expressed in subjective logic, which are later used for decision making (as discussed in Chapter 5).

4.1.2 Data in the World Model

The world model is a directed graph that contains vertices for the actual data, any known entities (i.e., other vehicles or RSUs), and *detectors* that determine the trustworthiness of either data or entities. The

[1]For performance reasons, one might imagine aggregation or history pruning techniques to remove or condense older information. We designed Maat with these ideas in mind, but consider their implementation and evaluation future work. Similarly, as with most VANET applications, the assumption is that this data is removed when the vehicle is reset, or when a trip ends. The implementation details of these aspects are outside the scope of this thesis.

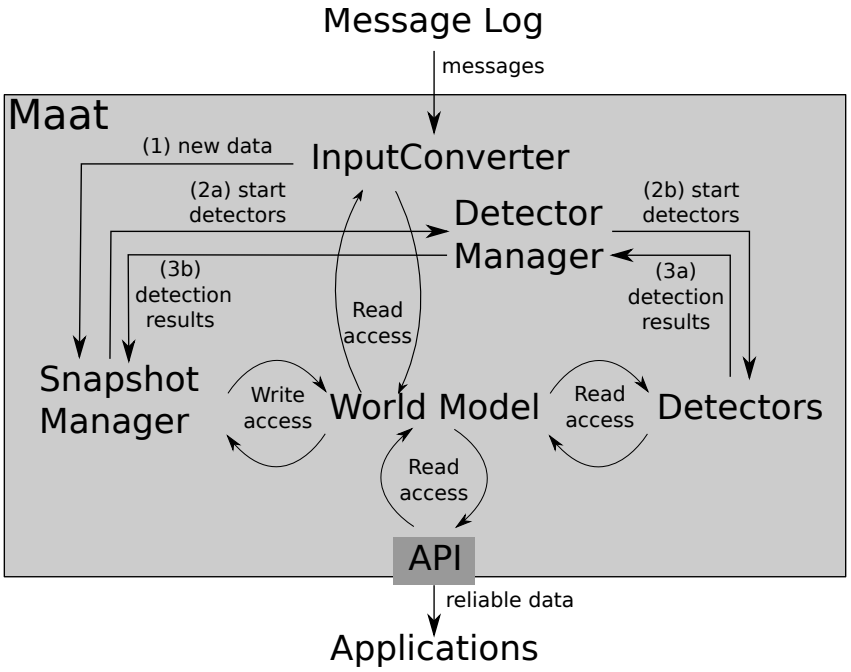


Figure 4.1: Maat's conceptual workflow

vertices for data are referred to as *data items*, which are vertices that contain information about a specific event (e.g., a decentralized environmental notification message warning about a traffic jam) or a series of related events (e.g., beacon messages from the same sender). We group closely related messages, such as beacons, because typical VANET applications rely heavily on such messages for core functionality, such as tracking nearby vehicles. In Maat, we thus maintain a history of messages in the same data item, rather than creating a separate data item for every message. However, before discussing the details of how these components interact, we first provide a more formal definition of a Maat world model:

Definition 1 (Maat world model)**.** *A world model* WM *is a directed graph* $G = (V, E)$*, where the vertex set* $V = V_{Det} \cup V_{Data} \cup V_{En} \cup \{\text{MDS}\}$ *consists of four subsets, describing for types of vertices. These are the representations of detectors* V_{Det}*, data* V_{Data}*, entities* V_{En} *and the decision node* {MDS}*. The edge set* E *is a set of directed edges that describe a relation between two vertices, labeled with a subjective opinion.*

The vertex types mentioned in Definition 1 also restrict which edges may exist: vertices with data (i.e., $v \in V_{Data}$) cannot have outgoing edges, and the decision node MDS may not have incoming edges. These restrictions are discussed in detail below. Edges represent subjective logic opinions between the vertices in this graph: the trust in data is then intuitively a set of paths between the decision node MDS and the data vertex in question. We place additional restrictions on the edges between vertices that are related to the semantics of those vertices; thus we first introduce the semantics and contents of the vertices.

Definition 2 (Data item)**.** *Data items* V_{Data} *represent individual pieces of information, such as beacons or data about specific events, and contain a history that represents previous values.*

The concept of using data items is an idea we originally proposed in [115]: data items represent claimed events, which may or may not be valid[2]. As in that early proposal, Maat's data items are *leaves*, in the sense that they have no out-going edges. Events can here refer to a broad spectrum of messages: either specific occurrences, such as traffic jams, objects or road conditions, or periodic behavior such as beacon messages. In the case of beacon messages, Maat considers beacons from the same sender to

[2]In our implementation, we assume that a beacon represents exactly one claim, namely that the information contained within is correct and consistent with previous beacons. It is in principle possible to split a beacon into multiple claims, e.g., one for a position, one for a speed, one for direction. This *granularity* represents a trade-off that will be addressed later in this chapter.

belong together (i.e., in the input converter in Figure 4.1), and stores the corresponding messages in a history list in the same data item. This history is ordered by world model version, as discussed in Section 4.2.2. In a data item, the following meta-information is stored:

- the relevant variable types
- a history that specifies the values this data item had at specific times
- the referent, i.e., to what or whom this data item refers (typically a road segment or vehicle)

Definition 3 (Entity)**.** *Entities V_{En} represent external sources of opinions, which are typically other vehicles that express opinions about each other or about specific data items. A vehicle can also express opinions about another vehicle's data (i.e., an entity can have an opinion about data without being its referent).*

The idea behind representing vehicles as entities relates to Raya's work, as well as our earlier proposal [115] and the general concept of trust networks. Specifically, we represent entities independent of the related data in order to assess their reliability separately. This allows us to essentially mirror a wide variety of node-centric mechanisms, which typically report trustworthiness through messages exchanged between vehicles (e.g., trust or reputation values). These can be modeled as opinions between entities; because entities can have opinions about arbitrary other opinions, this part potentially adds cycles to our graph. A detailed discussion of this issue will be provided in Chapter 5, which discusses trust and fusion.

Another important reason to separately represent entities in the world model is that this simplifies dealing with privacy issues. Although our discussion focusses on detection, and the entities represent vehicles, it is equally possible to use pseudonyms in their place. This makes Maat compatible with most privacy-preserving VANET proposals that have been made. Future work could even add a module to the world model that expresses relationships between entities, e.g., when two pseudonyms are determined to be the same entity. By annotating the world model with this extra information, it becomes possible to locally link pseudonyms, which potentially leads to a lower risk of Sybil attacks and fewer detection errors. An alternative way to use this information is to combine all pseudonyms from the same source in one entity, making Sybil attacks impossible; however, this requires an extremely reliable linking process and is likely only feasible in the back-end (as discussed by Bißmeyer et al. [89]). For the remainder of this thesis, we will focus on the case where linking is taken care of by a different component (possibly through queries on Maat's world model), which decides locally which messages originate from which entity. We will not address attacks that target this part of the system[3].

Definition 4 (Detector)**.** *Detectors V_{Det} represent misbehavior detection mechanisms, which can vary from a simple plausibility check to a complex vehicle tracking approach.*

Detector vertices are one of the major differences between Maat's world model and the multi-graph approach that we originally developed [115]. In the multi-graph approach, we allowed an arbitrary number of edges between the ego vehicle, the other vehicles and the corresponding data. Apart from the fact that it makes queries more complex to understand, the multi-graph approach is also needlessly complicated from a semantic perspective: the opinions are normally held by detectors, and not necessarily by the vehicle on which those detectors execute. We decided to avoid this complexity and instead represent each detector as a graph. This also allows us to express opinions between the decision node and the detector, versioning these in a consistent way; this was not possible in the previous model. Another advantage of representing detectors independently is that we can include multiple instances of the same detector with different parameters, and we can more easily communicate reproducible results to other entities. For example, a vehicle can transmit an opinion expressed by one of its detectors, and include that detector's state in the message. If the detector relies only on messages exchanged between vehicles, a receiver can validate the detection process before including the opinion in its own world model.

[3] It is still possible to attack this by, e.g., attacking the linking component. On the other hand, Maat in both concept and implementation also allows the implementation of Sybil detection mechanisms. These detection mechanisms cannot, in the current implementation, state that several entities represent the same node, but they *can* issue a low trust value on all entities. In our view, this approach is the correct one; parallel use of several identities over periods longer than foreseen by pseudonym change periods should be considered malicious.

Each detector vertex stores the following meta-information:

- detector type
- values for parameters, if any
- detector state, if necessary

The parameters for a detector represent initial constant values, such as thresholds, which are also used to identify the detector. This is useful when applying Maat for a parameter study of a detector, or when using two different parameter sets of the same detector for different purposes.

Definition 5 (Misbehavior detection system)**.** *There is one special node in our model, called MDS, which represents the decision making component of Maat. It has no incoming edges, and it is used as a starting point for every query.*

To make it easy to query trustworthiness of a specific data item, we use the MDS node as the starting point of all queries in our graph. By searching paths between MDS and a specific leaf (data item), we now obtain a visual intuition of how Maat uses this graph for queries. Each path represents some evidence of trustworthiness or distrust, depending on the opinions at the respective edges. The paths are first resolved into individual opinions, and those opinions are then combined to output a final decision. As a performance optimization, we also allow edges between the MDS and any data or entity, which we refer to as a *cached opinion*. This stores any previous output, and is removed from any path query (i.e., paths for evaluation must always have length > 1).

4.1.3 Edges

So far, we have explained the different types of vertices, but not the different relations between these vertices. In the simplest case, every edge between two vertices in this model is a subjective logic opinion expressing the opinion of the originator on the destination. For example, if a detector D expresses an opinion about a data item V, the edge contains an opinion ω_V^D, representing D's confidence in the correctness of V. A detailed discussion of opinions will be provided in Chapter 5; here we focus on the subtle differences in semantics between the edge types, dependent on the vertices involved.

Entity to Data edges mean that either an entity E has transmitted a specific opinion on the data V, or Maat has added a default opinion from E to V because E is the referent of V. These edges are used to represent how confident an entity is in its own results, which is particularly useful if the entity itself is uncertain about the transmitted data. This is especially common in aggregation scenarios, as discussed also by Dietzel in their thesis [128]; these edges allow seamless integration with their ideas.

Detector to Entity or Data are two types of edges that represent detector outputs. These opinions represent the core of the actual detection process: each detector can output an arbitrary amount of edges on different nodes every time that new information is received. These edges are sent back to the snapshot manager (see Figure 4.1), and subsequently stored in the world model. Note that detectors are somewhat trusted here; we allow any detector to arbitrarily generate edges on different entities and data. The idea is that this allows maximum flexibility, allowing a detector to revisit their issued opinions at any time.

Entity to Entity edges are Maat's representation of a traditional trust graph. This allows Maat to take advantage of previous results in this area, allowing vehicles to exchange trust values to deal with individual malicious vehicles over the long term. Maat's trust in these entities is independently computed by any node-centric detectors.

MDS to Detector edges represent Maat's trust in the performance of the detector. Without further information, this trust is assumed to be the same in all detectors; however, for technical reasons discussed in Chapter 5, it is useful to scale down the immediate output of each detector. This is exactly what these edges are used for. It allows for the design of a dynamic configuration component to Maat, which can choose weights for each detector at run time. This is useful, because our evaluation results in Chapter 6

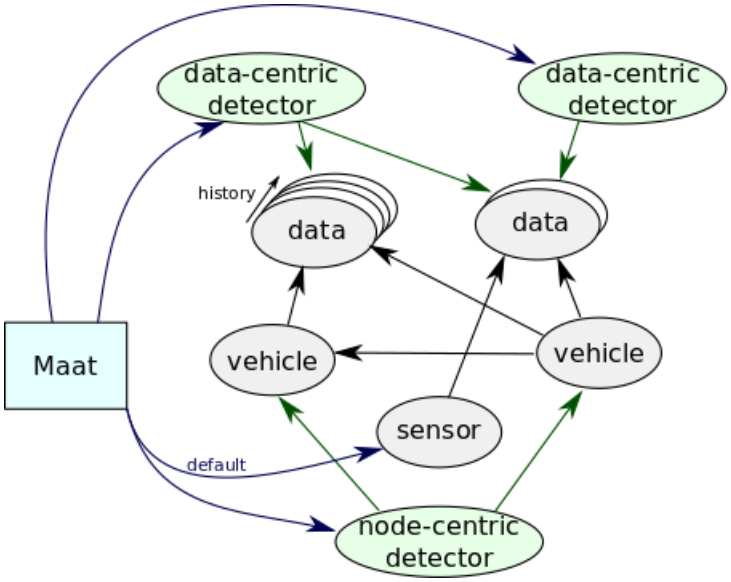


Figure 4.2: A Maat world model.

suggest that most detectors are sensitive to the context (e.g., type of road), which can be determined by a dynamic component. To understand this, consider a speed plausibility mechanism: the expected speed strongly depends on the type of road and the existence of other traffic. This has also been reported by others [29, 149].

MDS to Entity or Data edges are used for caching purposes; to optimize performance, the same result should not be computed twice. Basically, this means that when information about some data or an entity is queried, the resulting opinion is stored in one of these edges.

4.1.4 Summary

To summarize, the Maat world model consists of vertices and edges, who together represent the state of the world surrounding a particular vehicle. The leaves of the world model (i.e., vertices without outgoing edges) represent the actual data that is relevant to applications; querying for this data triggers a path search from the MDS to this data, which results in a trust evaluation of this data. This is essentially summarized in Figure 4.2; as pointed out here, data items also contain a history – this will be addressed in detail in the next section.

Note that this conceptual model is also useful for the communication and exchange of the data in the model. If a vehicle decides to report a specific vehicle, it only needs to take a sub-graph of its world model and transmit this to other vehicles. As long as this only relies on VANET messages, vehicles can completely reproduce the results (assuming the original messages are cached somewhere, with signatures included, as often suggested in the literature). Should internal sensors of the reporting vehicle be involved (i.e., be part of the sub-graph), a receiver of such a sub-graph can also re-write all the data from the sensor to originate from the reporting vehicle. We discuss these ideas in more detail in Section 4.3.

4.2 New Data

The entire workflow shown in Figure 4.1 consists of more than just the world model itself; it also has read and write operations. Rather than continuous read and write operations to the world model, writes are centralized in the snapshot manager. Each write is a consistent change that either includes an update of the world model (triggered by the reception of a message from an external source), or the addition of new detection results. Although it is theoretically possible to write the detection results to the world model immediately, this requires weakening the consistency model that is used for the world model (see Section 4.2.3). Instead, we require that each change be written to the world model separately, through the

snapshot manager. This provides an implicit logical ordering (i.e., the order of writes): as long as Maat receives the same data in the same order, it will produce the same world model state. Because historic information is often important for misbehavior detection, as well as applications in general, adding new data does not cause old data to be overwritten, but rather the new data is amended to the model. We explain how revisions of the world model are maintained in Section 4.2.2. However, before discussing versioning, we need to explain how Maat knows what data goes in what item, which is explained in the following section.

4.2.1 Linking

To decide what data is stored in which data item, it is necessary to choose the *granularity* of the data. This term comes from the field of in-network aggregation, and is used to reason about what data is aggregated to which extent. Similarly, Maat's world model will always increase in size if all beacon messages are kept; it thus could be reasonable to delete old messages. Deciding what old data is no longer relevant is a task quite similar to the challenges faced by aggregation applications, where the goal similarly is to retain as much information in as little storage as possible. The choice of data granularity is always a trade-off between data accuracy (i.e., retaining as much information as possible) and storage capacity or bandwidth usage (actually using and sending the data, respectively). In the case of Maat, we design the system to keep all relevant data, at least internally, until Maat is reset, for maximum reproducibility (i.e., maximizing the data accuracy).

In addition, Maat must also decide what data is stored in which data item specifically, and not just what data to keep. Recall that every edge to a data item contains an opinion about the correctness of that data; however, there is clearly a difference between stating that the position information in a data item is correct, and stating that the complete beacon message in which that position information is contained is correct. Throughout the development of Maat, we experimented with several ideas for grouping messages. From the aggregation point of view, a common approach would be to group messages by space-time segments, but for many applications this granularity is too low. Another consideration was that we would store every single field of a beacon message in a data item; logical operators could then be used to generate new data items as necessary. The advantage of the latter approach is that the edges explicitly capture exactly what a detector outputs. However, this makes fusion very complex: how should related data items be combined, and which detector outputs should be considered when deciding on the reliability of messages? Maat forces us to think explicitly about these challenges; in this work we focus on the case where all outputs are combined to evaluate a single message (i.e., a data item represents an entire beacon message). We focus on offering a significant improvement in this fusion that does not introduce additional complexities that raise more unsolved challenges.

Moreover, the required increase in storage and computational requirements that a detailed representation would entail are no longer reasonable for this domain. ETSI foresees between 6 and 12 highly dynamic fields in a CAM, including most required fields (heading, speed, driving direction, longitudinal acceleration, curvature, and yaw rate); as well as various optional fields (which includes acceleration control, lane position, lateral and vertical acceleration, and steering wheel angle). Even in the simplest case, this means storing each field $F \in \mathbb{F}$ individually requires 6 times more data items ($|\mathbb{F}|$); when also considering combinations of different fields (e.g., speed and heading are evaluated together), $2^6 - 1$ date items ($|\mathcal{P}(\mathbb{F})|$) are needed. This means that, with n detectors, up to $n \cdot |\mathcal{P}(\mathbb{F})|$ opinions will need to be stored. Even when assuming each detector outputs at most one opinion ($n = 1$), each message would require $2^6 - 1$ edges to be updated or created, and $2^6 - 1$ data items must similarly be modified. Since one of Maat's fundamental requirements is that evidence for detection results are stored, this means that with a message frequency f of (up to) $10Hz$ and k vehicles, storage required for a dense scenario corresponds to $10 \cdot k \cdot (2^7 - 2)$ objects per second. In high density settings, it can easily happen that $k > 200$; with a conservative estimate of 20 bytes per object, this corresponds to approximately 5 megabytes per second. When also considering optional fields, this quickly increases to around 328 megabytes per second, which clearly cannot be stored for any meaningful length of time.

Even when the amount of messages that need to be processed is much smaller, note that the time available for processing messages is $(f \cdot k)^{-1}$, i.e., in the range of milliseconds per message. Although fusion for each message can be expected to achieve this performance for an individual message, splitting this message into multiple data items also implies that multiple steps are required to fuse together the detection results, further increasing complexity. Because one of the primary results from our literature

survey (see Chapter 2) is that including more detectors generally detects more attack types, we here prefer to minimize the time required for fusion. We consider further increasing the expressiveness of Maat's world model as future work, which can enable increased performance for (relatively) niche applications such as detailed reporting and revocation, as well as aggregation applications. Our work thus concentrates on providing reliable short-term detection results, which are well-suited to the applications being introduced in the C-ITS domain in the near future.

4.2.2 Versioning

As new data comes into a world model, in many previous proposals it would replace or persistently modify the data previously stored there. The use of Kalman filters for tracking nearby vehicles is a good example: the update phase of the Kalman filter replaces the previous state with a new estimate of the position of a vehicle. Similarly, in a neighbor table of a routing algorithm, previous positions are normally discarded. However, this data may be quite useful in the detection process or during revocation. For this reason, Maat versions its world model, storing the historic, previous states in a data store for later retrieval. This works regardless of which data granularity choices are made; whatever changes would normally be overwritten after the data is stored in the world model gets replaced with an update operation.

Maat stores the different application states that correspond to each message in a history, because typically receiving a message just replaces the old information in the application. After this change is applied and detection is performed, the version represents a consistent snapshot of the vehicle's world view. In the world model that Maat uses, this process is implemented in a simplified way: for every write, a version counter is incremented, and each component in the graph (vertex or edge) has a history that maps the version to its data. To generate a view where only the latest data is available, one can simply ignore the history, and to go back to any previous world model state, one can simply use the version as a reference. This is possible because all write operations are centralized in the snapshot manager, which is the only thread that has write access to the world model. More precisely, Maat uses a single writer design, it can guarantee global consistency for applications that request data, even when the detection process would normally only provide eventual consistency. However, this requires that the entire detection process is in itself atomic: linking the data, doing the detection and writing the data to the model should be one atomic step. Although this provides the foundation necessary to provide globally consistent results, it requires that the snapshot manager waits before writing the detection results into the graph, and thus Maat's performance is limited by the slowest detector. In the next section, we will explore some ways that potentially reduce this constraint.

4.2.3 Consistency

Maat performs detection using multiple detectors that execute independently, and which are potentially implemented by third parties. With the wider availability of multi-core processor architectures throughout every field, at least the detection process itself, which is independent of other detectors, can be executed in parallel. In its current design, Maat places some restrictions how detectors behave in order to preserve consistency in the world model. In particular, writes are implemented through a standard locking mechanism (i.e., writes can only be performed once readers release their lock, and readers wait for the writer to finish). Because Maat atomically writes new data and new detection results into the world model, retrieving any version of the world model and performing read operations on this version will always provide the same view. In the detection process, Maat first links the data to existing data, updates the version, informs the detectors and requests their output, before modifying the world models' contents. The detector manager ensures that all detector outputs are together seen as a single atomic message by the snapshot manager (see Figure 4.1). Therefore, the sequential ordering of messages (at the snapshot manager) defines a sequentially consistent world model.

These restrictions forbid the modification of the world model itself, and thus restrict interaction between detectors during the initial pass over the message. It also reduces performance, since this model requires waiting for all detectors to finish. However, this is entirely reasonable for the type of detection typically performed by Maat: a model where the order of detector execution matters for the final decision is fundamentally flawed. Decisions reached by such a system would not be reproducible, and therefore not constitute evidence for misbehavior (or for correctness). Because this is our primary area of application,

and because we are not primarily interested by real-time constraints, Maat was implemented with this stronger consistency model.

Throughout Maat's development, we investigated several alternative consistency models and constraints on detectors. The goal was to provide a satisfactory model that both enables sufficient flexibility to implement real detectors, while at the same time allowing the results to be used by applications as fast as possible. Specifically, two alternatives were considered: a stateless model and a merging model. The stateless model required each detector to perform its detection in a stateless manner, which is also independent of pre-existing data in the world model. This corresponds closely to a stateless stream processing setup (as discussed above), and is very similar to every-day network security tools like firewalls. The merging model requires each detector to produce the same output given the same message and world model; a bijective translation from world model to detector-internal data representation was assumed for this model. This is also the model discussed in our earlier work [115].

The *stateless model* assumes all detection results depend exclusively on the incoming message itself. As messages are typically individual packets with known size and contents, these restrictions will generally limit the complexity of the detectors. The expectation was that plausibility and behavioral detectors would be compatible with these restrictions, based on our initial literature review. However, two developments made clear that this would not be the case: first, our focus moved to also include data received from the vehicle-internal bus systems, which Maat integrates seamlessly as if they were normal messages. Many plausibility mechanisms rely on such data for their checks (e.g., a vehicle's local position), and therefore a purely stateless implementation of such detectors is impossible. The other development was that plausibility checks often implicitly assume that previously received information is trustworthy, i.e., detectors need to be stateful. Examples of this include checking the plausibility of speed values by validating movement (i.e., computing the distance moved over the time between two messages and comparing it to the claimed speed).

The *merging model* was a much more flexible approach that we took after it turned out the stateless model was not going to be sufficiently reliable. In this model, Maat's processing overhead is minimal, and every incoming message is simply passed to each detector instantaneously. Although Maat with this model preserves an internally consistent world model, a translation function would filter everything except that data which is relevant to the detector. Using an inverse translation, a detector could then output results and store its state in the world model, making detectors much more independent processes. If any other messages have arrived in the intermediate time, Maat would inform the detector, and each of these could then revise their decision as needed. However, this model is very strongly dependent on the order of execution and the relative execution speed; this model does not necessarily even provide causal consistency. Depending on how the revision of decisions is implemented, this also requires faster detectors and generates additional load if many messages come in at the same time.

In Maat, the merging model is adopted; however, it turns out that this model has difficulties when messages come in faster than they can be processed by detectors. In this situation, the question arises whether (a) messages received before the current message are only visible after having been processed by *all* detectors, or (b) messages are instantly visible to all detectors and applications. Both options have advantages and disadvantages; for our current prototype, we implement (a), which means that the results will always be consistent, but it may take longer to process (since all detectors have to finish before starting the processing of the next message). In the case of (b), if a detector D_1 uses message M_0 in its detection process for M_1, before another detector D_2 has finished its detection of M_0, then the resulting world model may be inconsistent if D_1 incorrectly assumes M_0 is valid. For example, if D_1 decides M_1 is invalid *because* M_0 is valid, after which D_2 decides M_0 is invalid, the world model will only have evidence of invalid messages. In other words, the information that D_1 assumed M_0 is valid is lost. Because this is a common pattern in detection, we decided that (a) is the better option for our prototype.

One notable alternative we considered and discarded to resolve these challenges is a tree-structured versioning system. In this model, detection results can be written at any time, rather than waiting for every detector to finish, and they must contain a reference to a version associated with the read operations that were performed during detection. This reference was generated by giving every detector the same snapshot on which they perform their detection, giving the same view to every detector. This model also allows new incoming messages to be pre-processed and sent to every detector before they complete their detection, as described for the merging model. However, the writes now do not provide a consistent time line: thus a tree-structured versioning model is required to be able to reconstruct the world model

state on which a detection result is based. Although it is technically possible to generate merge messages for detectors, this greatly increases the implementation effort required for detectors. More importantly, initial experiments have shown that this model has much more significant performance overhead, due to the volume of extra messages generated. It also greatly increased the risk of live-lock occurrence, freezing the entire world model, due to implementation errors in a single detector.

Consistency in Maat

To make the entire processing chain reproducible, in that the same message added to the same world model results in the same new world model version, there are two important requirements missing from the system so far. The first is the assumption that all detectors included in the system are deterministic. This means that a detector will always output the same result given its state and the diff, i.e., change to the world model, which is defined by the previous state world model and the new message. To see why this is reasonable, consider a detector that does not satisfy this requirement: this detector could classify the same message differently under the same circumstances. Such probabilistic decisions can not be reproduced reliably, and therefore are not suitable for a multitude of goals that Maat aims for, such as revocation. Although future work could explore these types of decisions for other applications, where responsiveness is more important than complete precision, for this thesis we will assume determinism.

The second assumption needed for reproducibility depends on whether detectors use the results of other detectors. If each detection result depends only on the detector's internal state, the received message, and any previously received data[4], then no additional assumptions are required. In this case, a merging model could be used: every detector independently computes its results and the snapshot manager updates the world model graph by simply adding these edges (without additional synchronization). This enables immediate access to each detector's results as soon as they are written: the snapshot manager does not wait for completion of all detectors. However, there are a number of different detectors in the literature whose opinion depends on other detectors, for example, on which of two data items is more trustworthy. In those cases, it is necessary for the snapshot manager to wait for every detector to terminate before writing all results. If it would process new messages before writing the completed detection results, the termination order of detectors would cause the results to become indeterministic. Thus, in Maat's current design, processing is done strictly in-order: the linking, detection and writing operations are performed through a single write, and the next linking only starts after the previous write.

4.3 Misbehavior Reporting & Revocation

Maat maintains a history of received messages in the world model that applications can access. A natural next step is to enable a reporting process that uses this history of messages in order to report misbehavior to a certificate managing authority. In this section, we briefly discuss misbehavior reporting, explore how Maat could be used for evidence management, describe a message format adopted from Bißmeyer et al. [89], and discuss the limitations of this approach. This discussion is adapted from a recently published paper [157].

4.3.1 Misbehavior Reporting

To report attacks, Bißmeyer et al. [89] already defined a data sharing approach through misbehavior reports. These reports are created independently and signed by every vehicle that detects misbehavior. In essence, they consist of a list of suspicious vehicles that are to be reported, a list of nearby vehicles, a trust statement for each vehicle, and signed evidence. The evidence is a list of messages from the suspicious sender (e.g., a list of signed beacon messages). The trust statement consists of a trust value, a duration, a distance, and the first message received from the associated vehicle. Notice that there is no information regarding how the trust statement is arrived at. To limit the size of these messages, the authors suggest that the reporting node can choose to leave out neighbors to save bandwidth. The reports are transmitted to a central authority, where they are processed further.

A central authority, such as the misbehavior authority that is part of the proposed SCMS by Brecht et al. [168], is then responsible for handling the misbehavior detection and eventual revocation

[4]To be precise, this means only on entities, data items and edges between them.

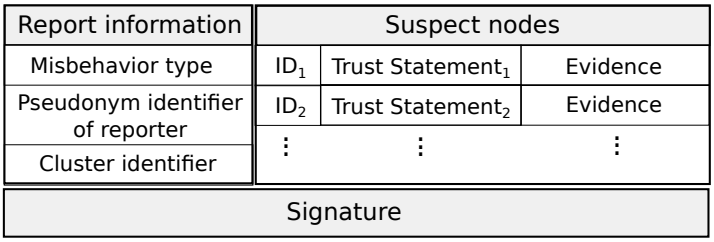


Figure 4.3: Sample structure of a misbehavior report

globally. Since reports may be delayed due to intermittent connectivity, this process is not suitable for immediate response, but rather designed to behave in a forensic way. Reports are signed using pseudonym certificates, enabling the authority to validate their integrity. Depending on the specific model of the authority, it may also be possible to resolve the pseudonyms to real identities before processing them, in order to protect against bad mouthing attacks. In the proposal by Bißmeyer et al. [89], the authority can also validate the signatures of the messages included in the report, but it cannot directly verify the correctness of the trust values or the data in the messages or the report. Bißmeyer et al. essentially describe a transitive trust mechanism that the authority uses to determine a final trust in each node involved in these reports. Note that the actual evidence is never used in this process, except for the validation of the reports. In this process, a lot of data-centric information is thus lost, which could be useful for detection by the authority.

4.3.2 Sub-Graphs in Maat

As discussed previously, whenever a query is performed within Maat to validate the trustworthiness of either data or nodes, a list of paths is generated that represents all the available information about this vertex. This includes not only the trust values associated with each vehicle, as in Bißmeyer's reports, but also the different detection results associated with each vertex. The sub-graph produced by combining the list of paths is essentially all the information that the reporter has about this vehicle. Furthermore, if all the properties of detectors that are necessary for consistency are satisfied, the authority can completely reproduce the detection given the evidence. This is particularly useful if the detection result of the reporter is very uncertain, since it enables the authority to re-use the limited information from several reporters to draw conclusions that would otherwise not be feasible. In our work in this direction, we developed a similar message format for misbehavior reports, which is illustrated in Figure 4.3.

In addition to being able to reproduce results, transmitting the entire sub-graph also enables easy integration at the authority. As long as detectors for distinct vehicles are given distinct names, the sub-graphs from different vehicles can be completely combined into a single graph. This single graph is then essentially a Maat instance running at the authority, which works in exactly the same way as Maat itself. This means that the authority can also execute additional detectors, such as those that rely on computational power not available in vehicles. The results can seamlessly be combined with results received from vehicles through the fusion process detailed in the next chapter. It may also be possible to use the sub-graph exchange to perform cooperative detection (i.e., reporting locally), but this is likely not feasible due to bandwidth requirements.

4.3.3 Limitations

This also brings us to the limitations that should be outlined with regards to sub-graph exchange. Although this approach provides a lot of information, this also necessarily implies that the message size will increase significantly. One can imagine optimizations similar to those that apply to Maat in general, such as pruning the graph by filtering out data with negligible effects on the results. However, the

sub-graph will still be too large, especially when multiple reports are generated with respect to the same vehicle, but due to different events.

Another potential issue is that of computational redundancy. If the authority would completely re-validate the detection process by generating the opinions of each vehicle based on the evidence, this checks the integrity of the report, but also repeats all computation already performed by the vehicle. One might argue that at this point, one might as well simply transport the evidence to the authority and do the detection there. Whether this is significant depends completely on the volume of reports that would occur in a system where the vast majority of vehicles is benign. It may be possible to optimize this by performing spot checks on report integrity: however, this potentially enables the revocation of benign vehicles, and thus should be used with care.

Finally, an important limitation is that any data that originates from the vehicles' sensors is not signed, and therefore difficult to include in reports. Although Maat will allow the exchange of this data, these parts cannot be validated, yet they are often the basis for high-quality detection output. It is imaginable that future systems may have in-vehicle sensors that sign their data in some way, but this raises a whole new spectrum of key management and privacy-related questions.

Although sub-graph reporting may be a viable and interesting approach for misbehavior detection in the future, much of its efficacy depends on the detection process. In particular, the volume of false positives, the amount of attackers in the network, as well as the bandwidth for future systems are parameters that are not clear at this time. Since these parameters are the primary factors for the reliability of a reporting procedure based on sub-graphs, we instead focus on research that advances the state of the art in this regard.

4.4 Prototype

For the evaluation of Maat, a prototypical implementation was designed. The Maat prototype used for this thesis is an open source project, implemented in Java 8, using the JGraphT library to represent the graph. The versioning is implemented through a property graph (a type of graph where vertices and edges can contain arbitrary key-value pairs), where the properties are included in a standard HashMap. Maat's development also involved a large volume of bug fixes and implementation effort for the subjective logic library; we discuss these changes in the next chapter. Implementation work was also done for individual detectors; these are shipped directly with Maat. Maat's source code is available on Github[5]; the implementation of specific components will be discussed in the respective chapter in which evaluation is performed. Note that Maat is currently a *research prototype*; the primary goal of this implementation is to show the feasibility of this design, and to provide a basis on which the evaluation part of this thesis could be based. As this evaluation is focused on detection performance, the entire software design is centered around evaluation, not around runtime performance or memory use. We consider it future work to assess the runtime performance of Maat: this dissertation aims to answer questions of general feasibility, not real-world performance.

4.4.1 Limitations

The prototype implementation has several limitations compared to the design described above. These limitations are briefly discussed in this section; their impact on results will be described in the evaluation part of this thesis.

Edge caching

One significant computational optimization that we propose but do not implement is the use of edge caching. As discussed previously, the idea behind edge caching is that we use edges between the MDS and the data and entities to cache intermediate detection results for every version. However, this conflicts with one of the main goals of the prototype, namely to compare different ways of performing data fusion (as will be explained in detail in the next chapter). We implemented these fusion operators in Maat, and the prototype outputs all different results that can be obtained with these different operators. Because we did not aim for runtime or memory performance as a primary goal, we simplified the prototype and do not implement this optimization, as it does not affect detection performance.

[5] `https://github.com/vs-uulm/Maat`

Aggregation & Dissemination

Although the initial ideas for Maat come from the field of secure aggregation, our development has taken a turn towards more concrete applications. Because the majority of real-world applications of VANETs are primarily centered around beaconing, we focus on this method of information transfer. Since beaconing is relatively well-studied, we were able to work with a broader spectrum of detection mechanisms. As discussed in earlier sections of this chapter, we provide various conceptual ideas that can be used to integrate aggregation behavior into Maat seamlessly.

The same applies for dissemination; Maat's API can be used to extract and transmit the desired aggregated information. However, the added value of such an application is quite limited; in particular, the lack of publicly accessible implementations would make evaluation of Maat practically impossible. Therefore, we instead focus on reliable detection of misbehavior on beacons: much of those results are used anyway to locally exclude malicious actors from an aggregation process [128].

4.4.2 Maat Prototype as Evaluation Toolkit

As by now discussed quite extensively, the primary focus of the Maat prototype is evaluation of detectors. Because of the way Maat executes its detectors, it is also suitable for the comparative evaluation of these detectors. With some minor modifications, Maat can be used as a framework for comparing misbehavior detection mechanisms, as well as the extensive analysis of the influence of various detector parameters (e.g., thresholds) on the detection performance of this detector and Maat overall. We extensively apply this process in the evaluation part of this thesis, and therefore briefly discuss the process here.

Figure 4.4 shows the overall evaluation process that can be used for detector studies. By simply exchanging the detectors and inputting the same message logs to Maat, one can easily generate precision-recall graphs that show the overall behavior of a detector for different thresholds. As far as we are aware, our dataset and evaluation flow is the first publicly available comparative study of detector performance. In particular, because Maat is open source, anyone can add a new detector, or extend the dataset with new attacks and test those attacks against the detectors in Maat.

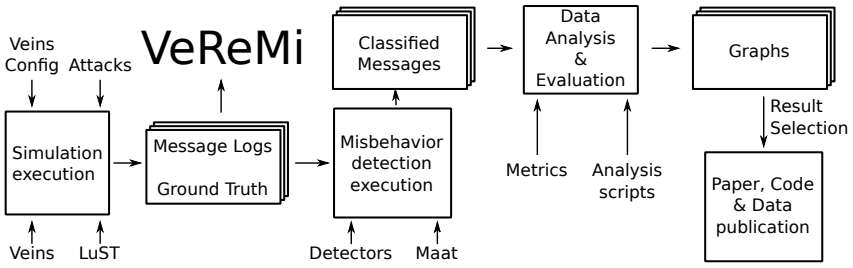


Figure 4.4: VeReMi's workflow – the detectors on the left side of the graphic can simply be replaced by new detectors; the simulation can be replaced by new simulations, resulting in an updated version of VeReMi.

Similarly, the attack implementations used for these experiments are public, and therefore this evaluation workflow can be used to design new attacks against any detection mechanisms that are publicly available. It is our hope that this evaluation toolkit functionality will enable scientific reproducibility of evaluation results. The workflow discussed here is part of our evaluation work, for which we designed a publicly available dataset: this dataset greatly reduces the amount of computational effort required to do reliable evaluations. We discuss this in detail in Section 8.1, where we introduce this dataset and show its application as part of the evaluation of the detectors we implemented for Maat. These detectors will be described in Chapter 6; however, we first describe how detector decisions are fused in the next chapter.

Chapter 5

Trust & Fusion

This chapter describes the details of fusion operations and standard trust mechanisms in subjective logic. These provide the foundations through which Maat combines the paths obtained by searching from the decision node to a specific leaf, performing misbehavior detection. As briefly described in Section 3.2.2, these paths are combined through subjective logic operations: traversal of paths corresponds to a transitive trust operator, while combination of paths can be performed through one of subjective logic's many fusion operations. In this chapter, we first focus on subjective logic, including several contributions that have previously appeared in [171]. We discuss the actual application of these operations in more detail after describing these contributions. Finally, we discuss the difference between trust and reputation, and sketch how existing trust schemes can be integrated into Maat.

5.1 Subjective Logic Fundamentals

This section provides the necessary fundamentals of subjective logic. This discussion is based on the book by Audun Jøsang, who originally described this logic framework for distributed trust management [144]. We only highlight the relevant details for our work, which is mostly based on the first five chapters, introducing the relevant mathematical details, as well as chapters 12 and 14, which respectively describe fusion and trust.

In particular, this section introduces the mathematical foundation of subjective logic and the hyper-Dirichlet model, which is an extension of the Dirichlet multinomial model. In addition, we introduce the respective notation in both settings, provide some intuition with regards to the interpretation, and describe a bijective map between subjective logic and the hyper-Dirichlet model.

5.1.1 Binomial Opinions

Binomial opinions [144, Ch 3.4], also referred to as subjective logic opinions or subjective opinions, express belief $\mathbf{b}_X(x) \in [0,1]$ over the possible values x of a random variable X, as well as an uncertainty u_X and a base rate $\mathbf{a}_X(x) \in [0,1]$ for every possible value. There is also an additivity constraint: $u_X + \sum_{x \in \mathbb{X}} \mathbf{b}_X(x) = 1$. In the binomial case, as the name implies, there are only two possible values in $\mathbb{X}$, often named x and $\bar{x}$ (or sometimes also true or false). T therefore the belief function $\mathbf{b}_X(x)$ is often replaced by a belief b and a disbelief d (representing $\mathbf{b}_X(x)$ and $\mathbf{b}_X(\bar{x})$, respectively), and the additivity requirement becomes $b+d+u = 1$, as discussed in [115]. Similarly, the base rate function $\mathbf{a}_X(x)$ is typically replaced by the base rate $a = \mathbf{a}_X(x)$, which implicitly defines the base rate of $\mathbf{a}_X(\bar{x}) = 1-a = 1-\mathbf{a}_X(x)$, since the base rate represents an *a-priori* probability (i.e., $\sum_{x \in \mathbb{X}} \mathbf{a}_X(x) = 1$). Note that binomial opinions are mathematically merely a special case of multinomial opinions, where $|\mathbb{X}| = 2$.

Definition 6 (Binomial Opinion)**.** *Let $X \in \mathbb{X}$ be a random variable over the binary domain $\mathbb{X} = \{x, \bar{x}\}$. A binomial opinion ω_X^A held by A over X describes the subjective assignment of belief by A to the outcome of X, consisting of an ordered triplet $\omega_X^A = (\mathbf{b}_X^A, u_X^A, \mathbf{a}_X^A)$. Here, $\mathbf{b}_X^A : \mathbb{X} \to [0,1]$ is the belief mass distribution over $\mathbb{X}$, $u_X^A \in [0,1]$ represents the lack of evidence, and $\mathbf{a}_X^A : \mathbb{X} \to [0,1]$ is the base rate distribution, with the additivity requirements that $1 = u_X^A + \sum_{x \in \mathbb{X}} \mathbf{b}_X^A(x)$ and $\sum_{x \in \mathbb{X}} \mathbf{a}_X^A(x) = 1$.*

An equivalent formulation is given by quadruplet of numbers, $\omega_X^A = (b_X^A, d_X^A, u_X^A, a_X^A)$, where $b_X^A = \mathbf{b}_X^A(x)$, $d_X^A = \mathbf{b}_X^A(\bar{x})$, $a_X^A = \mathbf{a}_X^A(x)$. The confidence of this opinion $c_X^A = 1 - u_X^A = b_X^A + d_X^A$ Due to the additivity requirement, storage of u_X^A is optional.

5.1.2 Multinomial Opinions

Multinomial opinions [144, Ch 3.5] express belief $\mathbf{b}_X(x) \in [0,1]$ over the possible values x of a random variable X, as well as an uncertainty u_X and a base rate $\mathbf{a}_X(x) \in [0,1]$ for every possible value. The conceptual purpose of multinomial opinions is to describe the available evidence or belief in each possible value $x \in \mathbb{X}$ in the domain of $\mathbb{X}$. We first provide a formal definition of multinomial opinions, before describing their application and interpretation with an example.

Definition 7 (Multinomial Opinion)**.** *Let $X \in \mathbb{X}$ be a random variable over the finite domain $\mathbb{X}$. A multinomial opinion ω_X^A held by A over X describes the subjective assignment of belief by A to the outcome of X, consisting of an ordered triplet $\omega_X^A = (\mathbf{b}_X^A, u_X^A, \mathbf{a}_X^A)$. Here, $\mathbf{b}_X^A : \mathbb{X} \to [0,1]$ is the belief mass distribution over $\mathbb{X}$, $u_X^A \in [0,1]$ represents the lack of evidence, and $\mathbf{a}_X^A : \mathbb{X} \to [0,1]$ is the base rate distribution, with the additivity requirements that $1 = u_X^A + \sum_{x \in \mathbb{X}} \mathbf{b}_X^A(x)$ and $\sum_{x \in \mathbb{X}} a_X^A(x) = 1$. Confidence is defined analogously to that of binomial opinions: $c_X^A = 1 - u_X^A$.*

If the random variable X is the outcome of tossing a potentially weighted die that uses colors instead of numbers, then its domain $\mathbb{X}$ contains the possible color values (e.g., red, green, blue, and so on). A multinomial opinion then assigns a *belief mass* to each color. Initially, the uncertainty u_X of this distribution can be set to 1, which is referred to as a *vacuous opinion*. The base rate of this opinion is the assumption that the die is not weighted, i.e., that all outcomes are equally likely: $a(x) = 1/|\mathbb{X}| = 1/6$ for all $x \in \mathbb{X}$. It is also possible to have opinions that assign all the available mass to belief, i.e., $u_X = 0$, which is referred to as a *dogmatic opinion*. This represents an edge case, since by definition zero uncertainty means that there is absolute certainty in the belief distribution, meaning the base rate has no bearing on the outcome at all.

To see how multinomial opinions can be applied, consider the die example above. By using opinions as evidence accumulators, each sampling of the random variable (i.e., throwing the die and adjusting the belief) increases the available evidence and thus the corresponding belief in the given outcome. The expected outcome of our opinion projects the belief and the uncertainty to a probability $P_X(x)$ by distributing the uncertainty over the potential values in $\mathbb{X}$ according to the base rate $a(x)$. More formally [144, Ch. 3]:

$$P_X(x) = \mathbf{b}_X(x) + \mathbf{a}_X(x) \cdot u_X$$

Multinomial opinions are often useful to represent evidence from a set of different sources $\mathbb{A}$, as common in trust management and data fusion. Referring again to the die example above: if the observers have some kind of color blindness, or the lighting conditions are sub-optimal, one can imagine that different observers have different opinions about the same observation. This is where multinomial opinions can be used by these observers to agree on the outcome of specific observations, as well as estimation of the real properties of the die (by observing repeated experiments). These operations can be achieved with fusion operations, as described in Section 5.2.1. One challenge for multinomial opinions is that it is not possible to represent an observer that cannot distinguish two colors in the above example (e.g., red-green color blindness), other than assigning the same amount of evidence to both values. However, this assignment implies that the relative frequency of red and green is the same, while in fact what we want to represent is the inability to distinguish between these variables. This is where hyper opinions are useful.

5.1.3 Hyper Opinions

Hyper opinions [144, Ch. 3.6] are the natural extension of multinomial opinions, which allow belief assignment to composite values. For example, taking $x_1, x_2, x_3 \in \mathbb{X}$, a multinomial opinion can assign a belief $b(x_1) = 0.1$, $b(x_2) = 0.3$, $b(x_3) = 0.4$, indicating the relative evidence between these three possible values. Some situations require the assignment of belief to composite values, e.g., $b(x_1 \cup x_2) = 0.4$, stating that there is evidence for either x_1 or x_2, without specifying the relative evidence between them. This is exactly the example we gave above with red-green color blindness, with $x_1 =$ red and $x_2 =$ green. Another such example was given by Hankin [50] in the context of Dirichlet hyper-Probability Density

Functions (hyper-PDFs), who uses the example of relative strengths of tennis players in a tournament. In their example, the outcome $X = x_i$ means i wins a tournament, and the belief assignment distributes the available evidence over the players. Assigning belief to $x_1 \cup x_2$ represents evidence that player 1 or 2 will win, but does not make statements about the relative probabilities between player 1 and 2.

Definition 8 (Hyper Opinion)**.** *Let $X \in \mathbb{X}$ be a random variable over the finite domain $\mathbb{X}$, with the power set $\mathcal{P}(\mathbb{X})$ and the reduced power set $\mathcal{R}(\mathbb{X}) = \mathcal{P}(\mathbb{X}) \setminus \{\mathbb{X}, \emptyset\}$. A hyper opinion ω_X^A held by A over X describes the subjective assignment of belief by A to the outcome of X, consisting of an ordered triplet $\omega_X^A = (\mathbf{b}_X^A, u_X^A, \mathbf{a}_X^A)$. Here[1], $\mathbf{b}_X^A : \mathcal{R}(\mathbb{X}) \to [0,1]$ is the belief mass distribution over $\mathcal{R}(\mathbb{X})$, $u_X^A \in [0,1]$ represents the lack of evidence, and $\mathbf{a}_X^A : \mathbb{X} \to [0,1]$ is the base rate distribution over $\mathbb{X}$, with the additivity requirements that $1 = u_X^A + \sum_{x \in \mathcal{R}(\mathbb{X})} \mathbf{b}_X^A(x)$ and $\sum_{x \in \mathcal{R}(\mathbb{X})} \mathbf{a}_X^A(x) = 1$. Confidence is defined analogously to that of binomial opinions: $c_X^A = 1 - u_X^A$.*

Remark 1 (Base rate over singleton values)**.** *Note that the base rate distribution is still over X here, meaning that X takes values from $\mathbb{X}$, not from $\mathcal{P}(\mathbb{X})$, and thus a hyper opinion has $(2^k + k - 3)$ degrees of freedom for a domain $\mathbb{X}$ of cardinality k, as discussed in [144, Ch. 3, p. 40].*

5.1.4 Dirichlet Distributions

Both multinomial and hyper opinions are directly related to Dirichlet distributions. The traditional Dirichlet distribution is defined by a vector of k parameters, α_X, resulting in the notation $\text{Dir}(\mathbf{p}_X, \alpha_X)$. This model has wide statistical applications; for our purposes, it will ground interpretations of opinions in a statistical sense. The idea is that the Dirichlet distribution represents a second-order probability distribution over a random variable X. This distribution is used to model the belief in each possible outcome. For the foundations of subjective logic, this probability density function (PDF) is adapted to an evidence-based formulation by setting the vector of parameters to represent the evidence for the possible outcomes of X. This is done by selecting a non-informative prior weight W and configuring the strength parameters α_X to be $\mathbf{r}_X + \mathbf{a}_X W$, where $\mathbf{r}_X(x) \geq 0$ for all $x \in \mathbb{X}$. This modification can then be used as a statistical basis for a subjective opinion, and is denoted $\text{Dir}_X^e(\mathbf{p}_X, \mathbf{r}_X, \mathbf{a}_X)$, where $\mathbf{r}_X$ is termed the evidence vector and $\mathbf{a}_X$ is the base rate distribution. The non-informative prior weight is set to $W = 2$, resulting in a uniform base rate distribution, as discussed in previous work [144, 160]. The PDF is then written as (compare [144, Ch. 3, p. 32-33]):

$$\text{Dir}_X^e(\mathbf{p}_X, \mathbf{r}_X, \mathbf{a}_X) = \frac{\Gamma\left(\sum_{x \in \mathbb{X}} (\mathbf{r}_X(x) + \mathbf{a}_X(x)W)\right)}{\prod_{x \in \mathbb{X}} \Gamma(\mathbf{r}_X(x) + \mathbf{a}_X(x)W)} \cdot \prod_{x \in \mathbb{X}} \mathbf{p}_X(x)^{\mathbf{r}_X(x) + \mathbf{a}_X(x)W - 1} \tag{5.1}$$

where $\mathbf{r}_X(x) + \mathbf{a}_X(x)W \geq 0$ and $\mathbf{p}_X(x) \neq 0$ if $\mathbf{r}_X(x) + \mathbf{a}_X(x)W < 1$.

This is the multinomial Dirichlet model, as also discussed in [160]. Hankin [50] formulated an extension of this model that allows evidence to be assigned to composite values, similar to hyper opinions, referred to as Dirichlet hyper-PDFs. Jøsang [144, Sec. 3.6.3] defines the evidence-based formulation of those hyper-PDFs, summarized as follows (where the superscript H clarifies that the domain is the hyper domain):

$$\text{Dir}_X^{eH}(\mathbf{p}_X^H, \mathbf{r}_X, \mathbf{a}_X) = \frac{\Gamma\left(\sum_{x \in \mathcal{R}(\mathbb{X})} (\mathbf{r}_X(x) + \mathbf{a}_X(x)W)\right)}{\prod_{x \in \mathcal{R}(\mathbb{X})} \Gamma(\mathbf{r}_X(x) + \mathbf{a}_X(x)W)} \cdot \prod_{x \in \mathcal{R}(\mathbb{X})} \mathbf{p}_X^H(x)^{\mathbf{r}_X(x) + \mathbf{a}_X(x)W - 1} \tag{5.2}$$

where $\mathbf{r}_X(x) + \mathbf{a}_X(x)W \geq 0$ and $\mathbf{p}_X^H(x) \neq 0$ if $\mathbf{r}_X(x) + \mathbf{a}_X(x)W < 1$.

Jøsang [144, Sec. 3.6.5] also describes the relationship between the PDF of the hyper probability distribution p_X^H and the probability distribution p_X: we refer interested readers to this work for a detailed discussion of the interpretative differences between these cases. For this work, it suffices that there is a well-defined grounding of both multinomial and hyper opinions in the Dirichlet multinomial model and the Dirichlet hyper-PDF model, respectively.

[1]The reason the reduced power set is used here is related to how hyper opinions are defined. The intuition is that a belief assigned to the empty set represents belief that the event X will not take place, while the belief in $\mathbb{X}$ represents belief that any of these elements may be the real value taken by X; this is intuitively the same as uncertainty.

5.1.5 Mapping Opinions to Dirichlet (H)PDFs

We now introduce the relation between opinions and Dirichlet (H)PDFs. There is a bijective map between Dirichlet HPDFs of the form $\text{Dir}_X^{eH,A}(\mathbf{p}_X^H, \mathbf{r}_X^A, \mathbf{a}_X^A)$ and hyper opinions $\omega_X^A = (\mathbf{b}_X^A, u_X^A, \mathbf{a}_X^A)$, as shown in Figure 5.1 (compare [144, Def. 3.9]). A similar map can be defined between multinomial opinions and Dirichlet PDFs, where the range of the mapping is constrained to $x \in \mathbb{X}$ instead of $x \in \mathcal{R}(\mathbb{X})$. Equivalently, for multinomial opinions and Dirichlet PDFs, the bijective map in Figure 5.1 applies where $\mathbf{r}$ and $\mathbf{b}$ for all composite values is defined as 0.

$$\left(\begin{array}{lcl} \mathbf{b}_X^A(x) & = & \frac{\mathbf{r}_X^A(x)}{W + \sum\limits_{x \in \mathcal{R}(\mathbb{X})} \mathbf{r}_X^A(x)} \\ u_X^A & = & \frac{W}{W + \sum\limits_{x \in \mathcal{R}(\mathbb{X})} \mathbf{r}_X^A(x)} \end{array}\right) \overset{\text{1-to-1}}{\longleftrightarrow} \left(\begin{array}{lcl} \underline{\text{For } u_X^A \neq 0\text{:}} \quad \mathbf{r}_X^A(x) & = & \frac{W \mathbf{b}_X^A(x)}{u_X^A} \\ 1 & = & u_X^A + \sum\limits_{x \in \mathcal{R}(\mathbb{X})} \mathbf{b}_X^A(x) \\ \underline{\text{For } u_X^A = 0\text{:}} \quad \mathbf{r}_X^A(x) & = & \mathbf{b}_X^A(x) \cdot \infty \\ 1 & = & \sum\limits_{x \in \mathcal{R}(\mathbb{X})} \mathbf{b}_X^A(x) \end{array}\right)$$

Figure 5.1: Bijection between hyper opinions (left) and Dirichlet HPDFs (right), as defined in [144].

For both the multinomial and the hyper opinion setting, there is an important edge case in this bijective map when the uncertainty is 0. As previously discussed, this semantically means that the belief distribution is "known", which implies there is infinite evidence. We briefly discuss the intuition behind this concept, as it is essential for the corresponding edge cases in the fusion operations, both in previous work [160] and in our formulations below. This is equivalent to taking the limit $\lim_{u_X^A \to 0}$ of the case for $u_X^A \neq 0$, giving the expression in Figure 5.1. To distinguish between different evidence variables' frequencies, the notion of relative degrees of infinity is used [52]. This intuitively corresponds to the relative likelihood of the different values for the variable X. For a detailed discussion of these conceptual notions, we refer interested readers to earlier work by Jøsang et al. [52, Sec. 4].

5.2 Fusion Operations

5.2.1 Multi-source Fusion

Multi-source fusion refers to the combination of opinions from multiple sources. Intuitively, this can be understood as a set of actors $\mathbb{A}$ coming together to agree on a common conclusion ($\omega_X^{\circ_{\mathbb{A}}}$), using some fusion operator $\circ$. This operator specifies precisely how to combine the information represented by each actor's subjective opinion. For operators that are both commutative and associative, $\omega_X^{A_1} \circ \omega_X^{A_2} \circ \omega_X^{A_3} \ldots$ is well-defined, and therefore fusion is straight-forward. However, the fusion operators defined as binary operators in [144, Ch. 12] are not all associative, i.e., in general, $\omega_X^{A_1} \circ (\omega_X^{A_2} \circ \omega_X^{A_3}) \neq (\omega_X^{A_1} \circ \omega_X^{A_2}) \circ \omega_X^{A_3}$, motivating the need to generalize the operators to accept arrays of opinions that are merged through fusion. This section defines multi-source fusion operations to generalize the operators not discussed in [160].

5.2.2 Cumulative and Averaging Belief Fusion

In previous work, the authors [160] propose multi-source fusion for the CBF and ABF operators. The definitions provided by the authors for CBF (Equations 16-19) and ABF (Equations 32-35) require some adjustments: the cases defined in the original paper are incorrect and lead to division by zero. Equation

16 represents the case where $\exists u_X^A \neq 0$, and states:

$$b_X^{\diamond(\mathbb{A})}(x) = \frac{\sum\limits_{A\in\mathbb{A}} \left(b_X^A(x) \prod\limits_{A_j \neq A} u_X^A \right)}{\sum\limits_{A\in\mathbb{A}} \left(\prod\limits_{A_j \neq A} u_X^A \right) - (N-1) \prod\limits_{A\in\mathbb{A}} u_X^A} \tag{5.3}$$

$$u_X^{\diamond(\mathbb{A})} = \frac{\prod\limits_{A\in\mathbb{A}} u_X^A}{\sum\limits_{A\in\mathbb{A}} \left(\prod\limits_{A_j \neq A} u_X^A \right) - (N-1) \prod\limits_{A\in\mathbb{A}} u_X^C} \tag{5.4}$$

Note here that if at least two dogmatic opinions are fused with any amount of non-dogmatic opinions, the case condition is satisfied. However, if $u_X^A = u_X^B = 0$ and $u^C \neq 0$, Equation 5.4 becomes:

$$u_X^{\diamond(\mathbb{A})} = \frac{0 \cdot 0 \cdot u_X^C}{(0 \cdot 0) + (0 \cdot u_X^C) + (0 \cdot u_X^C) - (3-1) \cdot 0 \cdot 0 \cdot u_X^C} = \frac{0}{0} \tag{5.5}$$

The equations for ABF have essentially the same issue. In general, division by zero occurs when at least one opinion has a non-zero uncertainty, while at least two (other) opinions have zero uncertainty: this leads to Case I in the original formulation, but in the corresponding equations, all the products in the divisions for $\mathbf{b}$ and u are zero. Therefore, a straight-forward correction is to require that *all* uncertainties to be zero whenever applying Case I. One should then also modify the condition for Case II accordingly, and additionally modify the equations such that any non-dogmatic opinions are discarded. This makes sense intuitively, since any non-dogmatic opinion represents finite evidence in the Dirichlet evidence PDF representation (through the bijective map of Figure 5.1), while a dogmatic opinion represents infinite evidence. Therefore, Case II discards finite evidence and essentially only combines the dogmatic opinions. The original authors have since published a revised version of the paper on-line [2], which corrects the error by adjusting the case definitions. We refer to their corrected article for the details.

Epistemic Cumulative Belief Fusion

Although it was not explicitly discussed in [160], a multi-source variant of epistemic cumulative belief fusion can also be derived from the aleatory form discussed above. The difference between aleatory and epistemic fusion is the type of knowledge represented by the opinion; the aleatory variant describes the fusion of information with respect to specific observations, while the epistemic variant describes the fusion of knowledge. For epistemic cumulative belief fusion (e-CBF) of multiple opinions (as opposed to aleatory CBF), the only reasonable approach is then to first apply the multi-source aleatory cumulative fusion described in [160], until all relevant epistemic knowledge has been cumulated. Only when a decision needs to be made should uncertainty maximization (as discussed in [144, Ch. 12]) be applied to the resulting opinion. To see why this is reasonable, consider that e-CBF is used in an on-line fashion, fusing two opinions together before fusing them with a third. Fusing an e-CBF result with epistemic knowledge from a third source skews results in favor of this third source, since the uncertainty controls how much of the belief mass of the new epistemic evidence is considered for the final result. Instead, one would normally expect that the three opinions are considered equally in the fusion process. Similarly, in practice, an analyst would first combine all the available knowledge (i.e., retrieve all relevant epistemic evidence from all sources), and only then make a decision – if more information becomes available later on, one would expect the analyst to re-do the computation, rather than fuse the uncertainty-maximized result with the new information. Thus, indeed, e-CBF should be performed by applying a-CBF to *all* opinions, and then the result should be uncertainty-maximized.

5.2.3 Belief Constraint Fusion

Belief constraint fusion (BCF) is the extension of Dempster's rule of combination to hyper opinions. Although Dempster's rule of combination is associative, this rule does not consider the base rate, as

[2] `https://folk.uio.no/josang/papers/JWZ2017-FUSION.pdf`

Dempster-Shafer theory does not foresee such base rate distinctions. If the base rate is not the same for all inputs, the operator defined by Jøsang [144, Ch. 12.2] is not associative, as pointed out by the author: we thus require a multi-source formulation of this operation. This is provided by the following definition:

Definition 9 (BCF of opinions for multiple sources)**.** *Let $\mathbb{A}$ be a finite set of actors and let $\omega_X^A = (\mathbf{b}_X^A, u_X^A, \mathbf{a}_X^A)$ denote the multinomial opinion held by $A \in \mathbb{A}$ over X. We define the belief constraint fusion of these opinions as the opinion $\omega_X^{\&\mathbb{A}} = (\mathbf{b}_X^{\&\mathbb{A}}, u_X^{\&\mathbb{A}}, \mathbf{a}_X^{\&\mathbb{A}})$ where*

$$\begin{aligned}
\mathbf{b}_X^{\&\mathbb{A}}(x) &= m^{\oplus\mathbb{A}}(x) \\
u_X^{\&\mathbb{A}} &= m^{\oplus\mathbb{A}}(\mathbb{X}) \\
\mathbf{a}_X^{\&\mathbb{A}}(x) &= \begin{cases} \frac{\sum_{A\in\mathbb{A}} \mathbf{a}_X^A(x)\left(1-u_X^A\right)}{\sum_{A\in\mathbb{A}}\left(1-u_X^A\right)} & \text{, for } \sum_{A\in\mathbb{A}} u_X^A < |\mathbb{A}| \\ \frac{\sum_{A\in\mathbb{A}} \mathbf{a}_X^A(x)}{|\mathbb{A}|} & \text{, otherwise} \end{cases}
\end{aligned}$$

Here, the belief mass is merged through Dempster's rule of combination [2, Ch. 3.1]:

$$m^{\oplus\mathbb{A}}(x) = (m_1 \oplus m_2 \oplus \ldots \oplus m_{|\mathbb{A}|})(x)$$

where

$$\begin{aligned}
(m_i \oplus m_j)(\emptyset) &= 0 \\
(m_i \oplus m_j)(x) &= \frac{\sum_{y\cap z = x} m_i(y) m_j(z)}{1 - K_{i,j}}\text{, for } x \neq \emptyset
\end{aligned}$$

and

$$K_{i,j} = \sum_{y\cap z = \emptyset} m_i(y) m_j(z)$$

Here, $m_i(x)$ refers to i's belief mass assignment to x in Dempster's setting (see also [2, Ch. 3.1]).

This definition essentially corresponds to the application of a map between the individual opinions to DST, applying Dempster's rule, and mapping back again. Although it is obvious that this is well-defined for the belief and uncertainty (since the translation is one-to-one [144, Ch. 5]), the base rate needs separate consideration. The special case where all the base rates are the same, as implicitly assumed by DST (i.e., that $\mathbf{a}_X(x) = 1/|\mathbb{X}|$), also holds trivially. However, subjective logic technically allows for different base rates: these are combined as a confidence-weighted average (first case in the definition above), as defined by Jøsang [144, Ch. 12]. This only works if there is at least one non-vacuous opinion (i.e., $\exists x \in \mathbb{X}, A \in \mathbb{A} : b_X^A(x) \neq 0$): if the confidences of all opinions are zero, the confidence is equal for all opinions, and therefore the most meaningful combination of potentially distinct base rates is computing their average. Note that both cases preserve the standard setting, where base rates are equal across all opinions (including the output).

5.2.4 Weighted Belief Fusion

Audun Jøsang defined the weighted belief fusion of two opinions in [144, Def. 12.8, p. 232], which we extend here to a multi-source variant as follows:

Definition 10 (WBF of opinions for multiple sources)**.** *Let $\mathbb{A}$ be a finite set of actors and let $\omega_X^A = (\mathbf{b}_X^A, u_X^A, \mathbf{a}_X^A)$ denote the multinomial opinion held by $A \in \mathbb{A}$ over X. Then we define the weighted belief fusion of these opinions as the opinion $\omega_X^{\widehat{\diamond}\mathbb{A}} = (\mathbf{b}_X^{\widehat{\diamond}\mathbb{A}}, u_X^{\widehat{\diamond}\mathbb{A}}, \mathbf{a}_X^{\widehat{\diamond}\mathbb{A}})$ as follows:*

Case 1: $(\forall A \in \mathbb{A} : u_X^A \neq 0) \wedge (\exists A \in \mathbb{A} : u_X^A \neq 1)$

$$
\begin{aligned}
\mathbf{b}_X^{\widehat{\diamond}\mathbb{A}}(x) &= \frac{\sum\limits_{A\in\mathbb{A}} \mathbf{b}_X^A(x)\left(1-u_X^A\right) \prod\limits_{A'\in\mathbb{A},A'\neq A} u_X^{A'}}{\left(\sum\limits_{A\in\mathbb{A}} \prod\limits_{A'\neq A} u_X^A\right) - |\mathbb{A}|\cdot \prod\limits_{A\in\mathbb{A}} u_X^A} \\
u_X^{\widehat{\diamond}\mathbb{A}} &= \frac{\left(|\mathbb{A}| - \sum\limits_{A\in\mathbb{A}} u_X^A\right)\cdot \prod\limits_{A\in\mathbb{A}} u_X^A}{\left(\sum\limits_{A\in\mathbb{A}} \prod\limits_{A'\neq A} u_X^A\right) - |\mathbb{A}|\cdot \prod\limits_{A\in\mathbb{A}} u_X^A} \\
\mathbf{a}_X^{\widehat{\diamond}\mathbb{A}}(x) &= \frac{\sum\limits_{A\in\mathbb{A}} \mathbf{a}_X^A(x)\left(1-u_X^A\right)}{|\mathbb{A}| - \sum\limits_{A\in\mathbb{A}} u_X^A}
\end{aligned}
$$

Case 2: $\exists A \in \mathbb{A} : u_X^A = 0$. *Let* $\mathbb{A}^{dog} = \{A \in \mathbb{A} : u_X^A = 0\}$, *i.e.,* $\mathbb{A}^{dog}$ *is the set of all dogmatic opinions in the input.*

$$
\begin{aligned}
\mathbf{b}_X^{\widehat{\diamond}\mathbb{A}}(x) &= \sum_{A\in\mathbb{A}^{dog}} \gamma_X^A \mathbf{b}_X^A(x) \\
u_X^{\widehat{\diamond}\mathbb{A}} &= 0 \\
\mathbf{a}_X^{\widehat{\diamond}\mathbb{A}}(x) &= \sum_{A\in\mathbb{A}^{dog}} \gamma_X^A \mathbf{a}_X^A(x) \\
\textit{where } \gamma_X^A &= \lim_{u_X^{\widehat{\diamond}\mathbb{A}}\to 0} \frac{u_X^A}{\sum\limits_{A'\in\mathbb{A}^{dog}} u_X^{A'}}
\end{aligned}
$$

Note that γ_X^A *is defined by this limit due to the bijective map (analogous to [144, Ch. 12] and [160]).*
Case 3: $\forall A \in \mathbb{A} : u_X^A = 1$

$$
\begin{aligned}
\mathbf{b}_X^{\widehat{\diamond}\mathbb{A}}(x) &= 0 \\
u_X^{\widehat{\diamond}\mathbb{A}} &= 1 \\
\mathbf{a}_X^{\widehat{\diamond}\mathbb{A}}(x) &= \frac{\sum\limits_{A\in\mathbb{A}} \mathbf{a}_X^A(x)}{|\mathbb{A}|}
\end{aligned}
$$

Remark 2 (Infinite evidence)**.** *In Definition 10, case 2 describes the resulting combined belief using the relative weight* γ_X^A *per actor with infinite evidence. This is analogous to the notion of relative infinities in the bijective map discussed in Section 5.1.5. We exclude all finite evidence parameters here, by only considering the actors with dogmatic opinions: all non-dogmatic actors will have finite (and therefore negligible) evidence.*

Remark 3 (No evidence)**.** *If no evidence is available (Definition 10, case 3) in any of the inputs, this operation would divide by zero (since* $u_X^A = 1$ *for all actors, making the nominator 0 in case 1). However, if no evidence is available, the fused opinion should obviously also have no evidence: this justifies the definition in case 3.*

We remark that confidence-weighted combination of base rates as defined in Definition 10 maintains the intuitive property that if all base rates of inputs are equal, the output has the same base rate, regardless of the case. Intuitively, since the base rate represents the belief in absence of information, this should always be the case, but the theory allows actors to have different base rates, in which case they are averaged.

This derivation should intuitively correspond to that derived from the Dirichlet HPDF, as shown in [144, Thm. 12.4] for two opinions. In short, this theorem derives that the weighted belief fusion

operator for two opinions is equivalent to confidence-weighted averaging of the evidence parameters of the two corresponding Dirichlet HPDFs. We should thus show that our definition corresponds to the confidence-weighted averaging of the evidence parameters of all opinions. This is defined as:

$$Dir_X^{eH}(\mathbf{p}_X^H, \mathbf{r}_X^{\widehat{\diamond}\mathbb{A}}, \mathbf{a}_X^{\widehat{\diamond}\mathbb{A}}), \text{where } \mathbf{r}_X^{\widehat{\diamond}\mathbb{A}}(x) = \frac{\sum_{A\in\mathbb{A}} \mathbf{r}_X^A(x)\cdot(1-u_X^A)}{\sum_{A\in\mathbb{A}}(1-u_X^A)} \tag{5.6}$$

Theorem 1. *Our definition of weighted belief fusion of opinions for multiple sources is compatible with weighted belief fusion of Dirichlet HPDFs.*

Proof. We want to show that the WBF of multiple Dirichlet HPDFs corresponding to subjective opinions, when mapped back to a subjective opinion yield the formulas in Definition 10. We show only the derivation of $u_X^{\widehat{\diamond}\mathbb{A}}$, where $u_X^A \neq 0$ for all $A \in \mathbb{A}$. The other derivations follow the same basic pattern and do not provide any additional insights.

Using the mapping between Dirichlet HPDFs and hyper opinions from Figure 5.1 we can compute the evidence of multiple fused HPDFs corresponding to hyper opinions as follows.

$$\begin{aligned}
\mathbf{r}_X^{\widehat{\diamond}\mathbb{A}}(x) &= \frac{\sum\limits_{A\in\mathbb{A}} \mathbf{r}_X^A(x)\cdot(1-u_X^A)}{\sum\limits_{A\in\mathbb{A}}(1-u^A)} \\
&= \frac{\sum\limits_{A\in\mathbb{A}}\left(\mathbf{b}^A(x)(1-u_X^A)\cdot\prod\limits_{A'\neq A} u_X^A\right)}{\sum\limits_{A\in\mathbb{A}}(1-u_X^A)\prod\limits_{A\in\mathbb{A}} u_X^A}
\end{aligned}$$

Again, using the mapping in Figure 5.1 we know that the uncertainty of the fused HPDFs is

$$u_X^{\widehat{\diamond}\mathbb{A}} = \frac{W}{W + \sum\limits_{x\in\mathcal{R}(\mathbb{X})} \mathbf{r}_X^{\widehat{\diamond}\mathbb{A}}(x)}$$

Plugging in $\mathbf{r}_X^{\widehat{\diamond}\mathbb{A}}(x)$ we can cancel W and receive

$$u_X^{\widehat{\diamond}\mathbb{A}} = \left(\sum_{A\in\mathbb{A}}(1-u_X^A)\prod_{A\in\mathbb{A}} u_X^A\right)\cdot\left(\sum_{A\in\mathbb{A}}(1-u_X^A)\prod_{A\in\mathbb{A}} u_X^A + \sum_{x\in\mathcal{R}(\mathbb{X})}\sum_{A\in\mathbb{A}}\mathbf{b}^A(x)(1-u_X^A)\prod_{A'\neq A} u^{A'}\right)^{-1}$$

Next, we can substitute $\sum_{x\in\mathcal{R}(\mathbb{X})}\mathbf{b}_X^A(x)$ by $1-u_X^A$. Now, one can split the sums $A\in\mathbb{A}$ into $|\mathbb{A}| - \sum_{A\in\mathbb{A}} u_X^A$, yielding the numerator as described in Case 1; the denominator can be obtained by appropriate re-ordering.

□

5.2.5 Consensus & Compromise Fusion

Consensus & compromise fusion (CCF) is conceptually designed to first achieve consensus, conserving the agreed weight of all inputs, and then compute a weighted compromise for the remaining belief mass based on the relative uncertainty and the corresponding base rates relative to intersecting sets. Because this second step generally results in a total distribution greater than 1, a normalization factor η is used, which is multiplied with the compromise belief to compute the final fused result. Let again $\mathbb{A}$ be a finite set of actors and let $\omega_X^A = (\mathbf{b}_X^A, u_X^A, \mathbf{a}_X)$ denote the multinomial opinion held by $A \in \mathbb{A}$ over X. Then, CC fusion $\omega_X^{\Subset\mathbb{A}}$ is defined as the result of the following three computation phases: 1) consensus phase, 2) compromise phase, 3) normalization phase.

Step 1: Consensus Phase

In the first phase, the consensus belief $\mathbf{b}_X^{\text{cons}}(x)$ is computed as a minimum common belief per x. The residual beliefs $\mathbf{b}_X^{\text{res}A}(x)$ of each actor are the differences between their belief and the consensus belief. The total consensus b_X^{cons} is the sum of all consensus beliefs.

$$
\begin{aligned}
\mathbf{b}_X^{\text{cons}}(x) &= \min_{A \in \mathbb{A}} \mathbf{b}_X^A(x) \\
\mathbf{b}_X^{\text{res}A}(x) &= \mathbf{b}_X^A(x) - \mathbf{b}_X^{\text{cons}}(x), \text{ for each } A \in \mathbb{A} \\
b_X^{\text{cons}} &= \sum_{x \in \mathcal{R}(\mathbb{X})} \mathbf{b}_X^{\text{cons}}(x)
\end{aligned}
$$

Step 2: Compromise Phase
The compromise belief $\mathbf{b}_X^{\text{comp}}(x)$ in a frame x is computed by adding four components: 1) the residue belief, weighted by all other actors' uncertainty, 2) the common belief in frame sets where the intersection is x, 3) the common belief in sets where the union is x, but whose intersection is non-empty, and 4) the common belief in the sets where the union is x and the intersection is empty. This can be formulated as follows:

$$
\begin{aligned}
\mathbf{b}_X^{\text{comp}}(x) = & \sum_{A \in \mathbb{A}} \mathbf{b}_X^{\text{res}A}(x) \cdot \prod_{A' \in \mathbb{A}, A' \neq A} u_X^{A'} \\
+ & \sum_{\substack{y_1,\ldots,y_{|\mathbb{A}|} \\ \text{s.t. } \cap_i y_i = x}} \prod_{i=1}^{|\mathbb{A}|} \mathbf{b}_X^{\text{res}A_i}(y_i) \cdot \mathbf{a}_X(y_i \mid y_j,\, j \neq i) \\
+ & \sum_{\substack{y_1,\ldots,y_{|\mathbb{A}|} \\ \text{s.t. } \cup_i y_i = x \\ \text{and } \cap_i y_i \neq \emptyset}} \left(\left(1 - \prod_{i=1}^{|\mathbb{A}|} \mathbf{a}_X(y_i \mid y_j,\, j \neq i) \right) \cdot \prod_{i=1}^{|\mathbb{A}|} \mathbf{b}_X^{\text{res}A_i}(y_i) \right) \\
+ & \sum_{\substack{y_1,\ldots,y_{|\mathbb{A}|} \\ \text{s.t. } \cup_i y_i = x \\ \text{and } \cap_i y_i = \emptyset}} \prod_{i=1}^{|\mathbb{A}|} \mathbf{b}_X^{\text{res}A_i}(y_i), \\
& \text{where } x \in \mathcal{P}(\mathbb{X})
\end{aligned}
$$

The second, third and forth terms of this equation taken together basically iterate over a tabulation of the different valid combinations for $y_i \in \mathcal{P}(\mathbb{X})$, and depending on the constraints defined under the sum, one of the three computations is made. Since set operations are associative and commutative, the ordering of $\mathbb{A}$ implied by this equation is irrelevant, since all orderings are included in the tabulation. This is analogous to the way this operator was initially defined by Jøsang [144].

Additionally, in the second phase the preliminary uncertainty mass u_X^{pre} is computed as the product of all uncertainties. The intuition is that this represents the common uncertainty all actors agree on. The value b_X^{comp} is computed as sum of the compromise beliefs over $\mathcal{P}(\mathbb{X})$:

$$
\begin{aligned}
u_X^{\text{pre}} &= \prod_{A \in \mathbb{A}} u_X^A \\
b_X^{\text{comp}} &= \sum_{x \in \mathcal{P}(\mathbb{X})} \mathbf{b}_X^{\text{comp}}(x)
\end{aligned}
$$

These values are used in the next phase.

Step 3: Normalization Phase
Since mostly $b_X^{\text{cons}} + b_X^{\text{comp}} + u_X^{\text{pre}} < 1$, a normalization factor η of b_X^{comp} has to be applied. We compute η as follows.

$$
\eta = \frac{1 - b_X^{\text{cons}} - u_X^{\text{pre}}}{b_X^{\text{comp}}}
$$

The belief on the entire domain $\mathbb{X}$ is then added to the uncertainty. The intuition behind this is that a belief in $\mathbb{X}$ is the belief that *anything* will happen. Referring back to the tournament example from

Hankin, the belief that $\mathbb{X}$ will win is the belief that a winner exists, without distinguishing between the singleton values in this set, which is equivalent to non-informative belief, i.e., uncertainty.

$$u_X^{\Subset\mathbb{A}} = u_X^{\text{pre}} + \eta \mathbf{b}_X^{\text{comp}}(\mathbb{X}).$$

After this transfer, we set $\mathbf{b}_X^{\text{comp}}(\mathbb{X}) = 0$.
Now, the belief can be combined over all $x \in \mathcal{R}(\mathbb{X})$ after applying the normalization factor.

$$\mathbf{b}_X^{\Subset\mathbb{A}}(x) = \mathbf{b}_X^{\text{cons}}(x) + \eta \mathbf{b}_X^{\text{comp}}(x).$$

Definition 11 (CC-Fusion of multiple sources)**.** *Given a finite set of actors $\mathbb{A}$ and their multinomial opinions $\omega_X^A = (\mathbf{b}_X^A, u_X^A, \mathbf{a}_X)$ over X for $A \in \mathbb{A}$. We define the resulting CC-fused opinion as a result of the previous three-step computation as $\omega_X^{\Subset\mathbb{A}} = (\mathbf{b}_X^{\Subset\mathbb{A}}, u_X^{\Subset\mathbb{A}}, \mathbf{a}_X)$.*

As this fusion operation is defined for subjective logic only, we do not prove it's equivalence to other operations in other models. However, we note that this operation always generates valid opinions, and that our definition reduces to the original definition for the case that $|\mathbb{A}| = 2$.

Remark 4 (Multi-source CCF always generates valid opinions)**.** *This holds if the output of multi-source CCF produces valid beliefs and uncertainties (in the range of $[0, 1]$), and the additive property holds, i.e., $\sum_{x\in\mathcal{P}(\mathbb{X})} b_X(x) = 1$. The first property holds, since the definition consists only of additions and multiplications of non-negative numbers, except for the definition $\mathbf{b}_X^{resA}(x) = b_X^A(x) - b_X^{cons}(x)$, which is at least zero, since the latter term is the minimum of all the beliefs in x. The second property holds due to the choice of η, and because $\mathbf{b}_X^A(\emptyset) = 0$ for well-defined belief functions:*

$$\begin{aligned}
1 \;\stackrel{!}{=}\; & u_X^{\Subset\mathbb{A}} + \sum_{x\in\mathcal{R}(\mathbb{X})} \mathbf{b}_X^{\Subset\mathbb{A}} \\
= \;& u_X^{pre} + \eta \mathbf{b}_X^{comp}(\mathbb{X}) + \sum_{x\in\mathcal{R}(\mathbb{X})} (\mathbf{b}_X^{cons}(x) + \eta \mathbf{b}_X^{comp}(x)) \\
= \;& u_X^{pre} + b_X^{cons} + \eta b_X^{comp} \\
= \;& u_X^{pre} + b_X^{cons} + 1 - b_X^{cons} - u_X^{pre} = 1
\end{aligned}$$

Remark 5 (Multi-source CCF is a generalization of the CC fusion operator)**.** *With $\mathbb{A} = \{A, B\}$, the steps resolve to these statements are as follows:*

$$\begin{aligned}
\mathbf{b}_X^{cons}(x) &= \min(\mathbf{b}_X^A(x), \mathbf{b}_X^B(x)) \\
\mathbf{b}_X^{resA}(x) &= \mathbf{b}_X^A(x) - \mathbf{b}_X^{cons}(x) \\
\mathbf{b}_X^{resB}(x) &= \mathbf{b}_X^B(x) - \mathbf{b}_X^{cons}(x) \\
b_X^{cons} &= \sum_{x\in\mathcal{R}(\mathbb{X})} \mathbf{b}_X^{cons}(x)
\end{aligned}$$

$$\begin{aligned}
\mathbf{b}_X^{comp}(x) = \quad & \mathbf{b}_X^{resA}(x) u_X^B + \mathbf{b}_X^{resB}(x) u_X^A \\
+ \quad & \sum_{\substack{y_1,y_2 \\ s.t.\ y_1\cap y_2 = x}} \mathbf{b}_X^{resA}(y_1) \mathbf{a}_X(y_1|y_2) \mathbf{a}_X(y_2|y_1) \\
+ \quad & \sum_{\substack{y_1,y_2 \\ s.t.\ y_1\cup y_2 = x \\ and\ y_1\cap y_2 \neq \emptyset}} (1 - \mathbf{a}_X(y_1|y_2)\mathbf{a}_X(y_2|y_1)) \mathbf{b}_X^{resA}(y_1) \mathbf{b}_X^{resB}(y_2) \\
+ \quad & \sum_{\substack{y_1,y_2 \\ s.t.\ y_1\cup y_2 = x \\ and\ y_1\cap y_2 \neq \emptyset}} \mathbf{b}_X^{resA}(y_1) \mathbf{b}_X^{resB}(y_2)
\end{aligned}$$

$$u_X^{pre} = u_X^A u_X^B$$
$$b_X^{comp} = \sum_{x \in \mathcal{P}(\mathbb{X})} \mathbf{b}_X^{comp}(x)$$
$$\eta = \frac{1 - b_X^{cons} - u_X^{pre}}{b_X^{comp}}$$
$$u_X^{\Subset \mathbb{A}} = u_X^{pre} + \eta \mathbf{b}_X^{comp}(\mathbb{X})$$

and finally, setting $\mathbf{b}^{comp}(\mathbb{X}) = 0$.
Except for differences in notation, this is the same as the definitions provided in [144, Ch. 12].

5.2.6 Example

Table 5.1: Extending the examples from [160]. The numbers presented here are rounded. The final row of the table presents the projection of each operation to the probabilistic space.

	Inputs			Fusion					
Parameters	A_1	A_2	A_3	aCBF	**eCBF**	**BCF**	ABF	**WBF**	**CCF**
$\mathbf{b}(x)$	0.10	0.40	0.70	0.651	0.442	0.738	0.509	0.562	0.629
$\mathbf{b}(\bar{x})$	0.30	0.20	0.10	0.209	0	0.184	0.164	0.146	0.182
u	0.60	0.40	0.20	0.140	0.558	0.078	0.327	0.292	0.189
$\mathbf{a}(x)$	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
$P(x)$	0.40	0.60	0.80	0.721	0.721	0.777	0.673	0.708	0.723

In this section, we extend the table from Jøsang et al. [160] with results for the fusion operations we have defined. This example considers a situation where three sources, $\mathbb{A} = \{A_1, A_2, A_3\}$, provide opinions over the same binary domain $\mathbb{X} = \{x, \bar{x}\}$. These opinions can be merged using the operators we discussed in this paper: the numerical results are shown in Table 5.1. We briefly discuss the functionality of each new operation (highlighted in bold in the table), and refer to the work by Jøsang for a detailed discussion of aCBF (referred to as CBF in their work) and ABF. The mathematical properties of all operators are discussed in the book that proposes them [144, Ch. 12], and we do not repeat such a discussion here.

- **aleatory Cumulative Belief Fusion** (aCBF) is suitable for situations where opinions represent independent sources of evidence. This operation is thus suitable for situations where different information sources with the same result imply there is more evidence for this fact. However, the assumption of independence means that this is not suitable for cases where actors are attempting to agree on a decision or achieve consensus.
- **epistemic Cumulative Belief Fusion** (eCBF) is the uncertainty-maximized version of aCBF. As previously mentioned, *epistemic* here refers to the fact that the opinions involved are used for representing knowledge, rather than documenting specific observations. Epistemic opinions are always uncertainty maximized: they represent exactly the available information about X beyond the base rate $\mathbf{a}$, and no more. This operation is useful for artificial reasoning about abstract events, which are not tied to a specific instance.
- **Belief Constraint Fusion** (BCF) is the generalization of Dempster's rule of combination that considers distinct base rates. The interpretation of BCF has been the subject of many works over the years [4, 144, 148], most importantly noting that conflict is essentially discarded.
- **Averaging Belief Fusion** (ABF) computes the raw average of evidence values from each actor. This operation is intended for cases where opinions are based on dependent evidence, e.g., different actors observe the same things but may establish different opinions. To fuse these, ABF can be used if all actors are considered equal. The most common example of ABF is a court in which a decision is reached by jury: each actor's opinion counts equally towards the final decision.

- **Weighted Belief Fusion** (WBF) is the evidence-weighted combination of belief from different sources. Similar to averaging belief fusion, it is useful where dependence between these sources is assumed. However, it introduces additional weighting by looking at the confidence of all sources, increasing the significance of sources that possess high certainty. This automatic weighting is useful for the combination of evidence from a large variety of uncertain sources which output high certainty in very specific scenarios. In such cases, ABF issues high uncertainty, while WBF outputs what is intuitively a consensus between experts: each expert individually choose their confidence, and thus their weight in the decision.

- **Consensus & Compromise Fusion** (CCF) is specifically designed to create vague belief from conflicting belief on singleton values. More precisely, generating a vague belief refers to the derivation of a belief of a composite value $x_1 \cap x_2$ if two opinions have high belief in x_1 and x_2 respectively. This is useful when different experts generate opinions identifying different options, such as when doctors with different expertise suggest potential causes in a diagnostic process. The fused opinion reflects the opinion of all experts, and illustrates the group as a whole is certain about a certain set of potential causes, without expressing relative likelihood.

5.2.7 Conclusion

As part of this thesis, we contribute multi-source fusion operations as non-trivial extensions of several existing fusion operators: the belief constraint fusion, the weighted belief fusion, and the consensus & compromise fusion. Since the native operators' definitions are non-associative, a definition is required to enable fusion of information from multiple sources. We also discuss the intuition behind the fusion process and the corresponding interpretation of the fusion result. Finally, we have proven that WBF corresponds to the confidence-weighted averaging of evidence in the Dirichlet multinomial model. These contributions will be applied by Maat as part of its fusion process; we provide all these extensions in order to be able to do a comparison of the different operations.

Finally, we would like to comment that both Dempster-Shafer theory [4, 95] and Subjective Logic [113] have been the subject of controversy over the years. In our work, we rely on the finalized description of subjective logic [144], which contains a clear map between subjective opinions and Dirichlet PDFs, resolving the main issue raised by Dezert et al. [113] (namely that different mappings have been proposed, and it is unclear which is the valid one). Nevertheless, the criticisms and alternative proposals made by these and other authors should be considered seriously. To our best understanding, the concerns raised have been resolved sufficiently, which is also evidenced by the continued study and extension of subjective logic at major publication platforms in the field [160]. We argue that our proposed derivations of multi-source fusion operations is a useful contribution to the field.

5.3 Trust

Computational trust is a multi-faceted topic that has a significant history both with respect to subjective logic and the existing work discussed in Chapter 2. In this section, we briefly discuss how Maat treats trust internally, by discussing these separate aspects, followed by a brief discussion of reputation. However, we point out that trust in ad-hoc networks is typically based only on previous interactions, and therefore existing mechanisms from the literature require well-performing detection mechanisms. For this reason, this thesis primarily focuses on fusion and trust over time (provided by schemes such as the exponential weighted average (EWA), see Section 5.4.2). This trust section therefore also describes how existing trust mechanisms fit into Maat.

5.3.1 Trust in Subjective Logic

Previous work [144, Ch. 14, Ch. 15] discusses the concept of computational trust in subjective logic. This includes a discussion of trust domains, which are the equivalent of the domain of opinions in the opinion types discussed earlier in this chapter, as well as an extensive discussion of the philosophical underpinnings and types of trust that exist. The particular distinction that is important here is the distinction between *referral trust* and *functional trust*. Referral trust describes how A's opinion about B describes how likely B is to provide accurate recommendations, while the functional trust of A in

B describes how likely it is that B will provide accurate predictions, according to A. The book on subjective logic [144] describes this with an example of car mechanic skill: if Bob states Eric is a good mechanic, this represents Bob's functional trust in Eric, while if Alice has referral trust in Bob, she might derive functional trust in Eric, even though she was not previously aware of Eric. Transferring this to Maat, we use referral trust to describe the decision node's confidence in detectors and a node-centric detector's trust in other vehicles, and we use functional trust between detectors and the data or entities they evaluate, as well as for entities that provide trust about other objects. Because in our setup, the entities are vehicles rather than humans, we assume that for opinions exchanged between vehicles about each other, the functional and referral trust can be conflated into a single value. Future work could theoretically distinguish these two trust types, but we expect the added value to be minimal, especially due to the ephemeral nature of these networks, as previously discussed by Raya [40].

As already mentioned, referral trust is used to represent recommendations. To resolve these recommendations to a derived functional trust (i.e., an opinion from Alice to Eric in the example above), subjective logic provides a trust transitivity operator. This operator is referred to as trust-discounting operator, because it basically takes the recommendation provided by Bob and scales it, depending on the existing referral trust in Bob. Mathematically, this means projecting the binomial opinion that describes trust in Bob, and then multiplying the result with the belief function; the uncertainty is then defined through the additive property, while the base rate remains unchanged. This operator is defined as [144, Def. 14.6, Def. 14.7]:

$$\omega_X^{[A;B]} : \begin{cases} \mathbf{b}_X^{[A;B]}(x) & = \mathbf{P}_B^A \mathbf{b}_X^B(x), \\ u_X^{[A;B]} & = 1 - \mathbf{P}_B^A \sum\limits_{x \in \mathcal{R}(\mathbb{X})} \mathbf{b}_X^B(x), \\ \mathbf{a}_X^{[A;B]}(x) & = \mathbf{a}_X^B(x), \end{cases}$$

This operator can be applied multiple times, if a path with multiple referral trust steps exists, until one arrives at functional trust. In other words, this operator can be applied repeatedly from left to right in a directed path in order to produce a single opinion representing the trust in intermediate steps. One important constraint required for this to work is that paths should be finite and contain no cycles. The issue of trust cycles is not addressed by subjective logic specifically; however, a broad variety of trust management and computational trust schemes exist that incorporate some kind of cycle removal algorithm. For some examples, we refer the interested reader to an overview by Victor et al. [84]. Within our prototype, we restricted the path length in our graph, essentially only allowing functional trust recommendations. This simplification was made primarily to simplify implementation – a straightforward solution to integrate all trust edges between entities would be to first query the trustworthiness of those entities evaluating the data of interest, and then use the cached edges in the fusion process.

5.3.2 Node-centric Detectors

Many node-centric detectors discussed in Chapter 2 and in our survey [170] can be adapted to operate in a subjective logic setting, similar to the way data-centric detectors can be adapted, as discussed in Chapter 6. Those adaptations work for the decisions made by detectors within a single Maat instance (i.e., internal to the vehicle). However, many node-centric detectors also require communication with other instances in order to work correctly. This is especially important for mechanisms that rely on node reputation (such as LEAVE and OREN), as well as mechanisms that perform any kind of event validation. In particular, in these two classes, messages are generated by a sending vehicle about the reliability of another vehicle, or about data received from another vehicle. These messages are intuitively similar to the opinions we have discussed so far; indeed, subjective logic was designed to represent such information exchange.

In our work, we have focused primarily on data-centric detection, as this is a necessary component in order to generate either of these message types. As previously mentioned and explained in detail in Section 4.3, we foresee that Maat's internal graph structure could be used for the direct information exchange between vehicles. However, it is also possible to integrate existing node-centric detection mechanisms, at least on a conceptual level. To be able to do this, all that is required is an implementation

of these mechanisms using subjective logic. This is usually easy, because most schemes use a probabilistic trust value, which can be converted into a dogmatic opinion with the corresponding belief/disbelief. In our model, we assume that there is only one way to issue opinions about other vehicles (i.e., the trust graph is not a multi-graph). This limits us to exactly one node-centric detector that exchanges information with other vehicles (i.e., node-centric trust mechanisms). This makes intuitive sense; if vehicles transmit multiple opinions about each other, it is very hard to decide which of these is valid. This also corresponds to how decision making works: in the end, one must either trust a vehicle or not.

5.4 Subjective Logic Extensions

This section describes additional extensions we made to subjective logic, which are primarily designed to evaluate Maat. This includes two operations that represent extreme cases in the decision logic; the MIN and MAJ operations, representing the minimum opinion (well-defined only for a binomial opinion) and the majority decision, which sides with the majority decision of its inputs (if any). Definitions can be found in Equation 5.7 and Algorithm 1 respectively. Essentially these operations are representative of naive fusion operations, as partially discussed in previous work. The purpose of these operations is to serve as a baseline against which to compare the other operators, as well as to demonstrate that dedicated fusion operators for subjective logic indeed perform better in practical situations.

In addition, we also define the computation of an *exponential weighted average* (EWA) for binomial opinions. This is useful when combining opinions from the same source at different time steps (i.e., different versions of Maat). Such combinations have been suggested as a means to "even out" classification errors made by individual detectors. For our evaluation of such mechanisms, a meaningful definition of the EWA for opinions was required; we provide the corresponding definitions here.

5.4.1 Baseline Operations

The MIN operator represents the most sceptical approach: each detector must explicitly accept the message, as defined in Equation 5.7. This operator is similar to fusion methods used in some related work in the past: for example, Stübing et al. [82] take this approach for their plausibility checks. Similarly, the designers of VEBAS [35] specify that all checks potentially detecting attacks must pass. Although this sounds like a reasonable approach, it only applies when the checks have a negligible false positive rate (i.e., falsely classify a message as malicious). It has been shown by these works, as well as other authors, that even seemingly obvious checks as the acceptance range threshold, often do not achieve negligible false positive rates [141]. Therefore, our expectation is that this operator will have a high false positive rate, which will lead to a low precision overall.

$$\omega_X^{MIN(\mathbb{A})} = \min_{A_i \in \mathbb{A}} (E(\omega_X^{A_i})) \tag{5.7}$$

$$\text{where } E(\omega_X^A) = b_X^A + u_X^A \cdot a_X^A \tag{5.8}$$

Another naive fusion operator is analogous to a vote: assume the majority of detectors provide a correct decision, and use this output. We implement this behavior through the MAJ operator, which computes the outcome of each input opinion through projection, and then outputs full belief or disbelief (i.e., a dogmatic opinion) to align with this outcome. In cases where there is no majority in either direction, it then seems natural to decide the outcome as uncertain (i.e., $(0, 0, 1, 0.5)$). This is algorithmically defined in Algorithm 1. Our expectation is that the MAJ operator will work very well for situations where detectors unanimously agree on the maliciousness of the message. However, detectors can be expected to be designed to detect known attacks, and it seems likely that MAJ will be vulnerable to messages that look partially legitimate.

5.4.2 Exponential Weighted Average

Trust Transitivity In addition to the different fusion operators, subjective logic offers trust transitivity to model trust relationships between entities, one of the purposes for which it was originally designed [144]. In our proposal, we use transitive trust to model the reliability of individual detectors

Algorithm 1: Procedure to compute majority opinion. Note that E(ω,x_i) refers to the projected probability of x_i given the opinion ω.

input : A list of opinions $\omega_X^{A_i}$
output: A fused opinion that represents the majority decision

$v \leftarrow 0$;
for $i = 0$ **to** $|\mathbb{X}|$ **do**
  $t_i \leftarrow 0$;
end
for $i = 0$ **to** $|\mathbb{A}|$ **do**
  find x_i s.t. $\mathtt{E}\ (\omega_X^{A_i},x_i) > \mathtt{E}\ (\omega_X^{A_i},x_j)$ where $i \neq j$;
  if *no such* x_i **then**
    $v \leftarrow v + 1$;
  else
    $t_i \leftarrow t_i + 1$;
  end
end
if $\exists! t_i$ *s.t.* $\forall j : j \neq i \implies t_j < t_i$ **then**
  return dogmatic opinion for x_i;
else
  return vacuous opinion;
end

in our system. Because detectors are designed to detect specific attacks (analogous to signature-based detection in IDS), and there is a wide variety of attacks, it is likely that many detectors coexist in a deployed system; hence, their reliable fusion is of the utmost importance. It is possible to directly apply fusion operations to detector outputs. However, this leads to the domination of the output by few, highly confident detectors: for example, a plausibility check with a hard threshold would always output a dogmatic opinion. Because fusion with a dogmatic opinion always results in a dogmatic output (since dogmatic implies infinite evidence), our earlier work [141] adapts existing mechanisms with hard thresholds to reduce the impact of this issue. We discuss these adaptations in more detail in Chapter 6, as they are of independent interest.

Here we propose an alternative to avoid dogmatic opinions all together: modeling the system confidence in a detector through an opinion, which is then combined with the detectors' output through trust transitivity. After this weighting process, the outputs of each detector are combined through fusion, as discussed in the previous section. The added advantage of this approach is that the system can be configured to temporarily disable certain detectors, e.g., in case they are known to perform poorly for specific scenarios. This process allows us to apply a dynamic weighting, which significantly weakens the assumptions that need to be made about these detectors, and thus it enables the inclusion of more attack-specific detection mechanisms.

An additional commonly used mechanism to reduce the impact of errors is the use of an exponential weighted averaging (EWA) [35, 88]. The idea is that classification errors tend to be random, and thus recent message history allows the correction of such errors by including the trust of those previous messages. Despite the risk of reputation or intermittent attacks, where the attacker only misbehaves for some messages, authors have suggested that this process is viable for reputation establishment. EWA is typically defined as follows:

$$T_t = (1 - \alpha)T_{t-1} + \alpha T_{\text{new}}$$

Where T_{new} is the new detector output, T_t is the trust at time t, and α is a weight parameter that assigns importance to newer values ($\alpha = 1$ discards old values completely). The choice of α determines the balance between removing classification errors (both false positives and false negatives) on the one hand, and on the other hand preventing attackers from impacting the system through a combination of malicious and benign messages. An analogous definition can be given in terms of the expectation (i.e.,

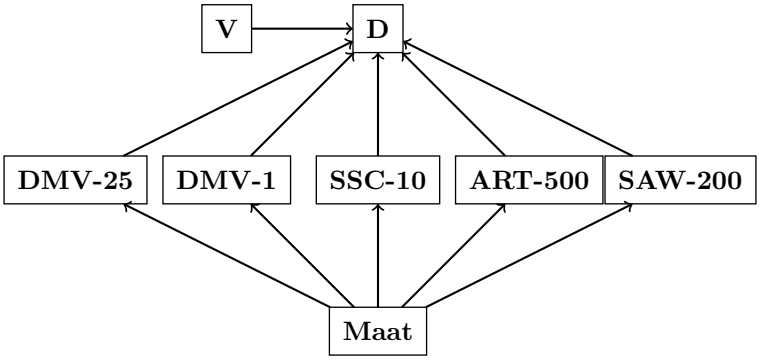


Figure 5.2: A world model with a single data item.

projected probability) of an opinion:

$$E(\omega_t^{\mathrm{A}}) = (1-\alpha) \cdot E(\omega_{t-1}^{\mathrm{A}}) + \alpha \cdot E(\omega_{\mathrm{new}}^{A})$$

Note that this definition implies there are multiple opinion choices possible for ω_t^{A}, because the projection from opinion to probability value fundamentally discards information. In our framework, we use EWA for each detector to incorporate past opinions on the message history before fusing the EWA adapted results. Thus, a choice needs to be made which value is chosen to represent the output in opinion space to fuse the results later on. Since our argument for the use of EWA is based on the assumption that errors in detector output are random, we choose the *uncertainty-maximized* opinion, which is the opinion with the correct expectation that has the most uncertainty. The result is used as input for fusion.

5.5 Fusion and Trust in Maat

Our system, Maat, takes the previously discussed components and builds a misbehavior detection framework that can be deployed with a dynamic set of detectors. Rather than simply combining fixed detection mechanisms, as proposed in previous work [35, 82], our approach is designed for flexibility in the combination of detectors, and the trust placed in them. The design of Maat is inspired by the previous work of Dietzel et al. [115], who also propose a graph-based representation of received data. As discussed previously, the world model consists of four vertex types: data items to represent received messages, entity vertices to represent senders, detector vertices to represent detectors, and a root node that represents the system's trust in detectors. The edges between these vertices represent expressions of trust through subjective logic opinions. An example of such a graph can be found in Fig. 5.2, where the data item D represents beacons received from the single sender, and the five detectors are those used in our evaluation. With this single leaf, every path shown in the figure is a path to the data item, and therefore used to evaluate the trustworthiness of the message. By default, no trust is placed in a sender (such as V in the figure), and therefore its opinion about its own message is not considered as input for trust.

To compute the trust in the current value of D, several factors are considered. First, the detectors execute their detection process independently, and produce opinions on the data as discussed in Chapter 6. When data in D is queried through Maat's API, the fusion process depicted in Figure 5.3 is applied. This figure shows all the steps applied by Maat to classify the data as either trustworthy or not, assuming the application only wants trustworthy data. First, a path search is done to find all existing output from the decision node, Maat, in Figure 5.2 to the queried data. This results in a list of paths between Maat, through detectors and possibly other entities, always ending in the data item from the query.

For evaluation, Maat can perform its evaluations either with or without an exponential weighted averaging (EWA) applied to the detection results. If the EWA is applied, then for each edge between detectors and data, the history of this edge is retrieved. This history contains the previous detection results for earlier messages that were linked to the same data item. These are then combined using the EWA, as discussed in Section 5.4.2. If no EWA is applied, then the system moves to trust transitivity with only the most recent opinions expressed by each detector.

Regardless of whether an EWA is applied, after this step, we still have a list of paths that needs to be reduced to a single decision. In the trust transitivity step, the *fusionability* parameter comes into

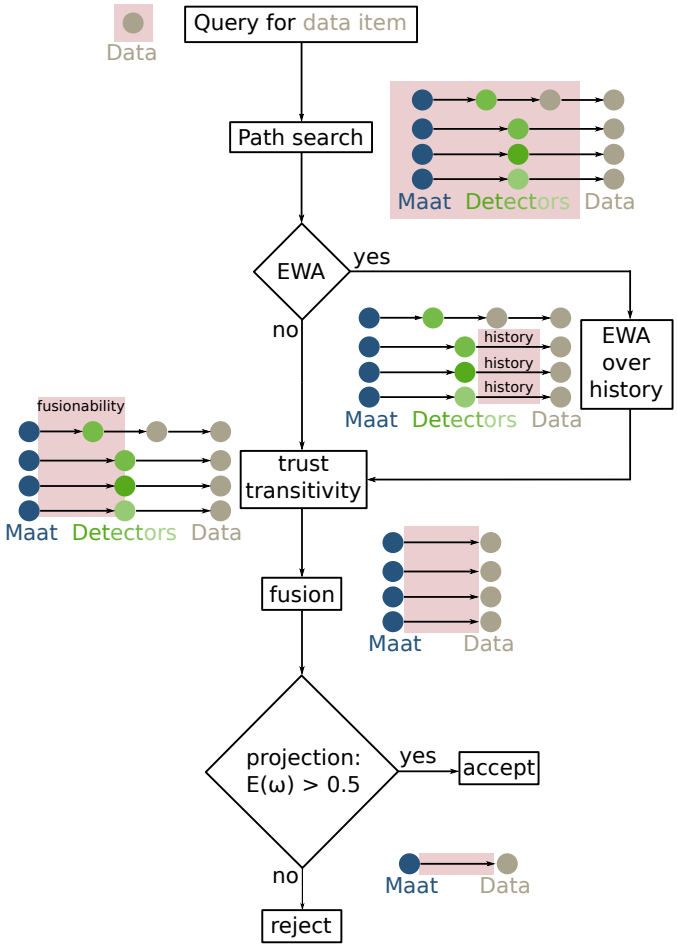


Figure 5.3: Maat's fusion process.

play; this weighs the influence of each detector. It is technically possible to choose different fusionability values for each detector (e.g., by applying some machine learning algorithm) to achieve the best detection results. Due to a lack of ground truth data, we chose the same fusionability for all detectors (i.e., all detectors are trusted equally), which means that the final weight in the fusion is dependent on the fusion operator. This results in a list of opinions between Maat and the data item D.

At this point, the actual fusion operation is applied; which operation is used can be configured in the prototype implementation of Maat. In our experiments we use this functionality to compare the different operations. Since the operations differ primarily in how the uncertainty is weighted in the fusion process, this should also be kept in mind during detector design. The result of this process is a single opinion of Maat on the data.

Finally, to decide on the trustworthiness of the queried data, the resulting opinion is projected. This operation maps the data to an expectation, a probability value that quantifies how likely the data is to be trustworthy (or not). If the expectation is over 0.5, Maat considers the data trustworthy and outputs that the data as true. If the expectation is equal to or below 0.5, the data is considered to be untrustworthy. Applications that can deal with the uncertainty expressed by subjective logic may also retrieve the opinion, rather than this binary decision. However, for our evaluation we require a final decision at some point: this decision approach appears to be the most reasonable.

Chapter 6

Detectors

This chapter describes detectors developed as part of the thesis and their integration into Maat. This includes both new detectors and re-implementations of existing ones. We also discuss a number of ways in which subjective logic can provide added flexibility for the detection process, and how detectors that are independent of subjective logic can still be integrated into our framework. We also provide proof-of-concept evaluations in Section 6.2, which are early results that motivated the development of Maat's fusion process in Chapter 5. A detailed evaluation of detectors and fusion can be found in Chapters 8 and 9. This chapter is based on several works that were published previously [139, 141, 172].

6.1 Detectors

This section introduces detectors implemented in Maat during this thesis, including both new detectors and re-implementations of existing ones from the literature. As discussed previously, we did not find a single detector throughout the literature for which source code was made public – we first focused on developing a baseline. To this end, we also re-implemented promising mechanisms from the literature, and designed alternatives where insufficient detail was provided to reproduce the work. We selected mechanisms that can always provide an estimate of accuracy for every message, using different models to validate the same data. The purpose of this is to show that the fusion discussed in the previous chapter can be used to boost performance by exploiting the availability of additional information.

In the following, detectors are labeled as either existing or novel. Existing detectors are re-implementations, whose purpose is to stay as close to the original design as possible. Some of the novel detectors are inspired by existing work, but they include modifications that make them more suitable for the generation of subjective opinions. This topic is addressed in the next section.

6.1.1 Existing Detector: Acceptance Range Threshold (ART)

The acceptance range threshold (ART) [29] is based on the assumption that transmission range is (approximately) fixed for all vehicles. Using this assumption, a detector can be derived that validates position information by checking whether the claimed position is close enough to the receiver. This also assumes that the receiver position is known and that errors in any position (e.g., due to the GPS of the sender or the receiver) are negligible. In essence, a reception area can be determined, which is the area from which received messages could have originated. If a position claim is received that is outside this area, it is an indication that this claim is false. If the position is in the area, the ART accepts the claimed position. An example can be found in Figure 6.1, where the receiver (green) and its reception area are shown. The attacker (red) claims to be at a distant position (grey), but this location is outside the receiver's reception range[1].

The ART can also be described as putting a simple threshold θ on the distance from which a received claim position is accepted. In this view, a claimed position at distance d is accepted if $d \leq \theta$. Note that an attack for any position *within* the reception range of a receiver cannot be detected by that receiver using the ART. The greater weakness of this detector is the assumption of a fixed transmission

[1]This detection could either imply that the attacker is attacking another vehicle near the grey position, or it may be an attempt to attack the green vehicle's routing algorithm to trick it to forward messages through the attacker.

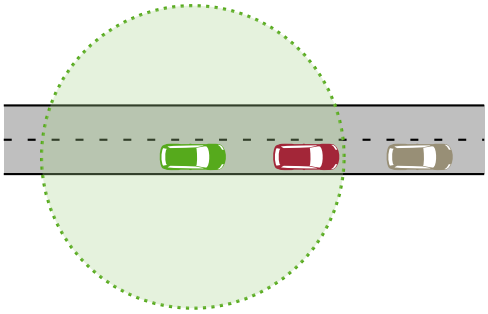


Figure 6.1: An example of the ART detector. Here, the green receiver vehicle assumes the indicated reception range, while the red attacker vehicle attempts to falsely claim the grey position.

range: in the real world, transmission ranges are limited by obstacles and various propagation effects. This can cause receptions outside of the predicted communication range for beacons according to current standards (which is assumed to be approximately 300 meters). These observations suggest that modeling the ART through subjective logic opinions provides a more adequate solution for fusion. We discuss these opinions in Section 6.2.1.

6.1.2 Existing Detector: Pro-active Neighbor Exchange

Because the ART cannot detect all attacks, its authors also proposed the pro-active neighbor exchange mechanism [29]. As the name suggests, the detector relies on the pro-active exchange of neighbor table information between all vehicles. The neighbor table contains the last received position for every nearby vehicle. The authors propose that this information can be compressed to a short identifier (e.g., a hash of the pseudonym certificate) for each neighbor.Using the position and neighbor lists for each neighbor, an attack can be detected by testing whether the new message is consistent with these neighbor tables. This is done by checking whether the potential attacker is present in each neighbor, as implied by the difference between the position of the neighbor and the claimed position. Similarly, it is checked whether the attacker appears in any neighbor tables that it should not appear in. Essentially, this is thus an indirect ART; the detecting vehicle only uses information from other vehicles.

An example of this detector is given in Figure 6.2. Here, the green detecting vehicle uses the positions of the blue and black vehicles, which were received at some previous time, to estimate their respective reception ranges. The position claim from the red attacker vehicle, marked in grey in the Figure, is now compared with these ranges. In this example, the attacker should appear as a neighbor in all three neighbor tables of the black vehicles, while not appearing in the neighbor table of the blue vehicle. However, the red vehicle is only in one of the three black transmission ranges, and thus will only appear in one of the three black position tables. Thus suggests that the grey position is false.

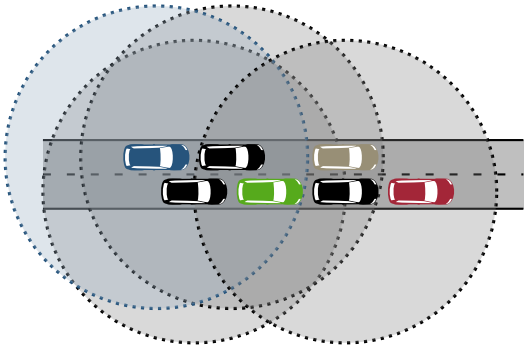


Figure 6.2: An example of the pro-active neighbor exchange. Here, the green vehicle uses a prediction of each neighbor's reception range to determine whether the red attacker vehicle should appear in the neighbor table of that neighbor. The attacker claims to be at the grey position.

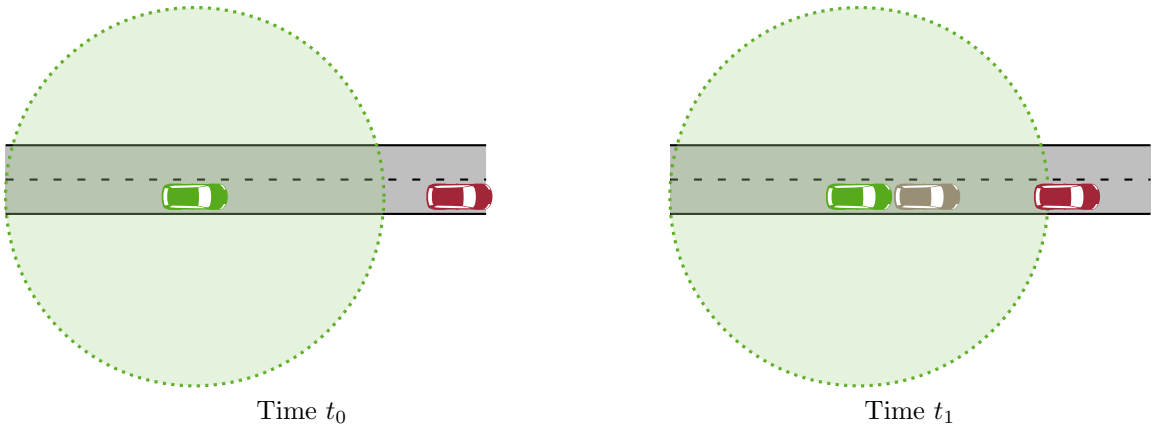


Figure 6.3: An example of the SAW detector. The green vehicle has not yet seen the red attacker at time t_0, but in the next beacon it receives from the attacker at time t_1, it claims a position directly in front of it.

Because mechanism relies on additional message contents, it is not standards-compliant, and the greatly increased message size can cause more message loss, while also reducing the bandwidth available for other applications. However, it enables the cooperative detection of attacks in the area near the detecting vehicle, where the ART is unable to detect attacks. For this reason, we chose the pro-active neighbor exchange as the complementary detector for the ART, with which we derived our first results for detector fusion. We discuss opinion generation for this detector in Section 6.2.2. The results of our first fusion experiments are outlined in Section 6.2.

The cooperative nature of this detector makes it susceptible to exploitation by more advanced attackers. For example, an attacker can use neighbor tables to falsely accuse various vehicles of misbehavior. Similarly, a successful false position claim on a detecting vehicle (i.e., a single false negative) with a well-chosen neighbor table can cause many errors (both false positives and false negatives) later on. This is because the detecting vehicle trusts all the accepted positions and corresponding neighbor tables. Finally, and most critically, the detector is highly susceptible to false positives, as also pointed out by the original authors [30]. The main causes of this are the fact that neighbor tables are designed to keep neighbors listed for some time, and the fact that timing is not accounted for. Because the detector assumes that all neighbor tables used in detection are up-to-date and recent, this leads to many errors in high-speed scenarios. For these reasons, in addition to the overhead caused, we replaced this detector with alternatives during the evaluation of Maat itself.

6.1.3 Existing Detector: Sudden Appearance Warning (SAW)

The sudden appearance warning (SAW) [30] is a detector that is based on the fact that vehicles do not suddenly appear out of nowhere. Instead, when approaching another vehicle, beacons are typically received at a regular interval. This means that position claims from new neighbors that are very close by are inherently suspicious. This plausibility check was considered particularly useful in earlier work [30], where the primary goal was to warn drivers of suspicious activity. The check potentially detects an important class of attack, which aims to trigger an immediate response from a vehicle by claiming a position directly in front of it. If this behavior is performed from the first contact, the SAW is one of the few mechanisms capable of detecting such an attack. This is illustrated in Figure 6.3.

The SAW detector can also be viewed as a threshold detector: if a newly appearing vehicle is closer than a threshold distance θ, then it is untrustworthy. This threshold depends largely on the communication characteristics of the network, in particular on how likely it is that messages between the sender and the detecting vehicle may be lost. This also points to a potential way for an attacker to exploit this detector: either through intelligent jamming or by simply jamming the entire channel for some time, the SAW can trigger on legitimate vehicles. The detector is also not suitable for situations where vehicles may actually appear suddenly, e.g., from a parking garage or parking spaces next to the road. To some extent, a poorly configured threshold may also lead to detection surrounding crossings, since buildings are likely to block signals until the threshold is violated. However, the SAW advantages for weaker

attacker models, where the attacker does not know whether its target is already aware of its presence. For example, an attacker may attempt to avoid detection by the ART by always claiming positions close to its target; the SAW can detect such an attack if the attacker assumes the target is already aware of its presence. Because it detects very specific attacks that cannot be detected by other detectors, we have integrated this detector into Maat. Due to the similarities with ART, we set the SAW's subjective logic opinions in the same way as the ART, except that values over the threshold are considered trustworthy and below the thresholds are considered untrustworthy. The expression for this is thus the one given in Section 6.2.1, but with the $\geq$ and $<$ symbols in the conditions swapped.

6.1.4 Existing Detector: Overlapping Position Detector (OPD)

Maat's recent releases also include a re-implementation of the data-centric part of the detection proposed by Bißmeyer et al. [45], which is based on the fact that two vehicles cannot occupy the same space. This detector, the overlapping position detector (OPD), detects false positions by checking for overlapping position claims using simple geometry. In order to actually detect most attacks, the authors proposed to compute additional rectangles, representing possible vehicle positions in the presence of GPS errors, based on the (claimed) speed of the vehicle. Since those rectangles are outside the actual position of the vehicle, an absolute statement (i.e., vehicles may never overlap) is insufficient, and some degree of trust is needed. In their work, Bißmeyer et al. used an additional trust mechanism on top of this proposal. In our implementation, we are interested in the detector itself, and thus we attempted to reproduce their work. To ensure that we specifically validate the position and make attacks more easily attributable, we compute the speed of each receiver from separate position claims. Other than that, our implementation aims to closely follow Bißmeyer's work; we also assume a 2D surface for vehicles (i.e., height differences are not considered) and compute the overlap result when a message arrives. A simplified example of a detection can be found in Figure 6.4.

Figure 6.4: An example of the OPD detector, where the innermost rectangles of the grey and green vehicle overlap. Since the innermost rectangles of vehicles correspond to the vehicle shape, we only show the vehicles in this figure. In this situation, the detecting vehicle is the only other vehicle, and we assume that this vehicle trusts itself, thereby classifying the claimed position as malicious.

A naive implementation of this detector requires many different parameters. Following Bißmeyer et al [45], we choose a maximum rectangle width (2.5 meters), as well as the same length and width for all vehicles (5 and 2 meters, respectively). In reality, the vehicle dimensions are stored in certificates, and thus the information necessary for the scheme is available[2]. Further factors that are part of the scheme are the amount of rectangles used in the detection process (3 rectangles), a factor $\alpha = 1.0$ that modifies vehicle length, and a factor that is used to determine how strongly speed influences the length of each rectangle ($d_s = \frac{1}{1.25}$) The values mentioned here are in line with Bißmeyer et al.'s choices. The parameter γ, which is an exponential parameter used to manipulate the certainty of the final detection result, is set to 0.5.

Because the output produced by Bißmeyer et al. is probabilistic in nature, it is natural to convert their output directly to an opinion. From their work it is clear that the message is considered malicious if an overlap is detected, but the lack of an overlap does not imply that the message should be accepted. Therefore, it is suggested that the opinion should be defined as follows:

$$\omega_{\text{OPD}} = (0, \frac{1}{(r+1)^\gamma}, 1 - \frac{1}{(r+1)^\gamma}) \tag{6.1}$$

Here, r is the smallest pair of overlapping rectangles, which are numbered from the innermost ($r = 0$, no overlap allowed) to the maximum ($r = 2$, some overlap possible). This reflects the relative unlikeliness that rectangles of the size specified by the above parameters actually interact. Similar to Bißmeyer et al. [45], we have not yet provided a thorough analysis of the relative behavior of all these

[2]It is possible that detection may not behave fairly if vehicles have different sizes, though.

parameters, but rather we conserve their values. Initial experiments [152], attempting to reproduce the positive results of these authors in our scenario against our attacks, suggest that this detector is not useful in detecting the majority of attacks implemented in our system. This applies even when the decision threshold for the opinion is modified (i.e., allowing full uncertainty to count as an accepted message). These results suggest that there is some bias in either the choice of attack and simulation scenario, or in the experimental design of Bißmeyer et al. [45] and the dataset that they use. Since none of these components used by Bißmeyer et al. are publicly available, we found that it was not possible to reproduce their results. Instead, we find that the detector performance is so poor that essentially none of the attacks we implement can be detected at all. Since it does not make sense to include a detector in fusion of which it is clear that no attacks are ever detected, we decided to exclude this mechanism from our evaluations based on these results.

6.1.5 Novel Detector: Distance Moved Verifier (DMV)

The distance moved verifier (DMV) is a simple detector, inspired by the minimum distance moved (MDM) [30], but designed to meet the requirements for immediate decision making. As the name implies, this detector validates whether the target has moved a threshold distance θ between two position claims. The MDM detector from Leinmüller et al. has similar properties; however, this detector is designed to wait for a certain amount of time before a decision is made (within which potentially many beacons can arrive). The DMV is a simpler detector that is aimed at immediate decision making as is desired within Maat. The DMV detects attacks and faults that cause lead an attacker to transmit the same position repeatedly, mirroring the purpose of the MDM. Although the primary purpose of both is to detect road-side attackers, an attack as in Figure 6.5 can also be detected. Note that the MDM would not have been sufficient to detect this example attack at t_1. The attacker would be aware of the threshold the green vehicle would use to operate the MDM, and thus the red vehicle could simply claim a valid position at some later time t_2 to avoid detection. This is not possible for the DMV.

Figure 6.5: An example of the DMV detector. In this example, the green and red vehicles move with the same speed, but the red vehicle claims the same position at both t_0 and t_1 (marked in grey in the latter), while in reality it has moved (shown in red).

A weakness of the DMV is that it is more susceptible to errors caused by normal traffic behavior. For example, vehicles in a traffic jam can trigger the DMV because they are not moving. Similarly, vehicles stopped for a traffic light can also trigger the DMV to distrust this vehicle. This behavior could be compensated for by changing this detector's fusionability in these specific traffic situations, which can be recognized easily. Independent of that, in Maat, the idea is that fusion can successfully merge the DMV with other results in such a way that this has a very low impact. Given the distance between two beacons d and a threshold θ, we configure the opinion as follows:

$$\omega_{\text{DMV}} = \begin{cases} (0.5, 0, 0.5), & \text{if } d \geq \theta \\ (0, 0.5, 0.5) & \text{if } d < \theta \end{cases} \tag{6.2}$$

6.1.6 Novel Detector: Simple Speed Check (SSC)

The simple speed check (SSC) is a detector inspired by the Kalman filter proposal from Stübing et al. [108]. Recall that a Kalman filter is a linear state estimation algorithm that is often used to track objects over time. This filter represents the state of each observed object as a vector of position and speed information, and essentially applies Bayes' rule to update the predictions and its error over time. If the object moves linearly, then the error converges. Stübing et al. proposed that the Kalman filter could be combined with some plausibility checks to validate new beacon messages. In this case, the prediction

of the Kalman filter is used to determine the trustworthiness of the new message[3]. If the message is plausible, the filter update process is applied; otherwise, the message is discarded.

The idea behind the SSC is to reduce the complexity of the Kalman filter; the required parameter choices (e.g., initial states and estimates of measurement and movement errors) could not be made based on Stübing et al.'s publications. As with the DMV, our implementation ensures that immediate decisions can be made, rather than waiting for a number of messages to pass before evaluation. This allows a potential misbehavior response algorithm to react in time, which is not possible when waiting for a number of message receptions. In this case, the computational complexity is also reduced significantly. The SSC computes an opinion based on three parameters; a base uncertainty ($U = 0.1$ in our implementation), a deviation that is considered acceptable ($\text{OK} = 0$ in the simples case) and a deviation that is considered completely untrustworthy ($\text{BAD} = 10$). The opinion is computed by comparing the extracted difference Δv between two messages, resulting in the following equation:

$$\omega_{\text{SSC}} = (1 - U - \frac{\Delta v - \text{OK}}{\text{BAD} - \text{OK}}, \frac{\Delta v - \text{OK}}{\text{BAD} - \text{OK}}, U) \tag{6.3}$$

6.1.7 Novel Detector: Doppler Shift Detection

In a recent work [139], we identified that channel characteristics could be used to design novel misbehavior detection schemes. We here describe one such mechanism, which is based on the Doppler shift in the signal propagation path that occurs when the relative velocity of two objects changes. The Doppler shift can be used to validate the velocity claims within a message, by comparing the relative velocity observed in the signal to the relative velocity derived from the message and the own velocity. These velocities should be approximately the same, but some deviation may exist; the velocity in the message must be measured before transmission, and the time between these events may lead to minor differences. We use an opinion to model this output, which can then be combined with other velocity-related detectors, such as the mobility grade threshold [30].

We now introduce the concrete scheme tested by al-Momani et al. [139]. First, note that the observed frequency of the signal in a single path is given by the following equation at the receiver side:

$$f = \left(1 + \frac{\Delta v}{c}\right) f_0 \tag{6.4}$$

where c is the propagation speed in the medium (i.e., speed of light), f_0 is the transmit frequency, which is assumed to be known to both the sender and the receiver, and Δv is the relative velocity between receiver and sender. The receiver can easily compute this from the claimed velocity and its own velocity[4]:

$$\Delta v' = v_r - v_s \tag{6.5}$$

Note that velocity can be expressed as a combination of speed $|v|$ and direction with respect to the positive x-axis θ:

$$v = |v| \angle \theta^\circ \tag{6.6}$$

This means that each velocity vector has a component in the x, y, and z axis. Restricting to 2D coordinates for flat roads, the relative velocity becomes:

$$\begin{aligned} \Delta v &= (v_{r_x} + v_{r_y}) - (v_{s_x} + v_{s_y}) \\ &= |v_r| \cos(\theta_r) - |v_s| \cos(\theta_s) + |v_r| \sin(\theta_r) - |v_s| \sin(\theta_s) \end{aligned} \tag{6.7}$$

Here, v_r, θ_r, and the observed frequency f are known to the receiver, allowing them to calculate the actual relative velocity Δv based on f. The difference in the claimed relative velocity $\Delta v'$ and the Δv computed from the Doppler shift measurements can then be used to validate the claimed velocity. The greater the difference between these sources, the more likely it is that the received beacon contains false information. By setting a hard threshold, the Doppler shift detector can easily be included in existing verification frameworks. However, it is also possible to derive a subjective logic opinion on the message through the difference $\delta = |\Delta v - \Delta v'|$.

[3] This is the opposite of a normal Kalman filter; that would use the message to update the filter state

[4] The sender can similarly estimate this using a previous message from the receiver.

Attacking Doppler Shift

Although Doppler shift is an inherent property of signals, an attacker can in principle still manipulate their signal to be consistent with their message. Here we briefly discuss such an attack and its limitations. Assuming the goal of the attacker is to convince receivers of a false velocity transmitted in its message, the attacker may manipulate the transmitted signal (e.g., transmit frequency) to avoid detection by our scheme. The attacker will thus use a manipulated transmit frequency $f_0^{'}$, computed by some algorithm $\mathbf{A}$ that determines the frequency corresponding to the desired speed $\Delta v_{desired}$, as:

$$f_0^{'} = \mathbf{A}(f_0, v_r, v_s | \Delta v_{desired}) \tag{6.8}$$

In a scenario with one receiver, this receiver would calculate Δv_{actual} based on the observed frequency, which will match the calculated velocity Δv derived from the beacon, assuming that such an algorithm $\mathbf{A}$ exists. However, in most scenarios, there is more than one receiver in the network. Notice that in this attack, a broadcast message will lead to the other vehicles in the network to detect the attack, since there will still be a mismatch between the false and computed velocities at any other receiver. A successful attack thus requires that only the target vehicle receives the message. This could in theory be achieved through highly directional antennas, which can then be used to simultaneously transmit multiple falsified messages to different receivers. Because vehicles are constrained to roads, and roads typically have multiple vehicles on them, it is very likely that a third vehicle will be between the attacker and the victim, or directly behind the victim. The third vehicle will then detect the attack even if the attacker uses highly direction antennas. A scheme using Doppler shift can thus be highly effective if the measurements are sufficiently accurate. However, the hardware required to perform such measurements may not be available in all vehicles.

6.2 Enhanced Detection

In this section, we present the results of our first integrated detection approach [141], the results of which suggested that the approach taken by Maat is feasible. We discuss the results and their implications in the following; we have integrated these detectors into Maat, which is evaluated as a whole in Part III of this thesis. We have re-implemented a number of existing detectors from the literature, as we were unable to find *any* detectors with public source code. These are the previously presented acceptance range threshold (ART), as well as the pro-active neighbor exchange. To study what would become Maat's approach, we applied the *consensus operator* from subjective logic, which is equivalent to the *cumulative fusion* discussed in Chapter 5. As part of Maat's evaluation in Chapter 9, we use these results for initial parameterization. In that chapter, we also evaluate the other fusion operations that were discussed in Chapter 5.

6.2.1 Enhanced ART Detector

A main disadvantage of the ART detector developed by Leinmüller et al. [30] is that it can only detect attackers that claim a location that is outside of the predicted transmission range of a receiver, while that receiver can still receive the message. The detector will inevitably suffer from false negatives – those cases where the attacker is in range, and transmits a false position within the transmission range of the receiver. There is also a significant degree of false positives related to actual transmission range. This degree of false positives comes from the fact that transmission range is not fixed, but rather changes depending on properties of the channel and obstacles in the vicinity. In order to resolve these weaknesses of the detector, fusion with other data sources is necessary. To enable this fusion, we need to convert the detection result of the ART detector into an opinion.

The idea behind the ART detector [30] essentially assumes a unit disc graph model for the transmission range, which cannot be considered realistic in any real-world setting. Thus, we propose that the opinion, which we need for fusion anyway, better represents the actual transmission range. We should thus select a high belief for nearby positions, high disbelief for positions far out of our transmission range, and high uncertainty for positions around the edge of our transmission range. To implement this, we chose a Gaussian distribution for the uncertainty u, with a mean of the expected transmission range (i.e., the ART) and a configurable standard deviation σ that should reflect the overall uncertainty about the potential transmission range. We normalize the uncertainty s.t. $u = 1$ if the measured δ equals the

threshold θ and choose the certainty to be $1-u$, reflecting that we have evidence for the message to be trustworthy. The certainty is assigned to belief when $\delta \leq \theta$ and to disbelief when $\delta > \theta$. This opinion can now be fused with, e.g., an opinion about the sender or other data-centric sources.

Notice that the approach presented here can easily be generalized to other variables for which the receiver can compute an independent estimate that can be compared with the message contents. In particular, for any estimated value δ, mean θ and variance σ^2, we can compute:

$$\omega = \begin{cases} (1 - e^{-\frac{|\delta-\theta|^2}{2\sigma^2}}, 0, e^{-\frac{|\delta-\theta|^2}{2\sigma^2}}) & \text{if } \delta \leq \theta \\ (0, 1 - e^{-\frac{|\delta-\theta|^2}{2\sigma^2}}, e^{-\frac{|\delta-\theta|^2}{2\sigma^2}}) & \text{if } \delta > \theta \end{cases}$$

6.2.2 Pro-Active Neighbor Exchange

As noted by the original authors [30], the pro-active exchange of neighbor tables suffers from high false positives. In order to improve the overall detection accuracy of the system, we improve this detector's output for increased flexibility. As in the enhanced ART detector, we mapped the output to subjective logic opinions, which represent the degree of belief, disbelief, and uncertainty. In particular, the output contains an uncertainty that is inversely proportional to the amount of available information.

Upon receiving a new beacon, the receiver computes the distance from the new position information to each neighbor and decide whether the sender of that beacon should be in the neighbors' neighbor table or not. Instead of making the decision at this point, as in the original detector approach, the receiver constructs two observations; *sender is benign* and *sender is not benign*. These correspond to whether the prediction is confirmed – for example, if the sender should appear in a neighbor's neighbor table and it appears, then this is considered as an occurrence of the *sender is benign* observation. Counting the total amount of mismatches x, we then compute an opinion as follows:

$$\omega = \left(\frac{\beta}{n} e^{-\frac{x}{10}}, \frac{n-\beta}{n} e^{-\frac{x}{10}}, e^{-\frac{x}{10}} \right),$$

where β is the amount of benign samples, and n is the total amount of samples.

Based on this detector, if the receiver does not have any neighbor, he will be totally uncertain about the newly received position information if it is correct or not. The more neighbors, the more certainty in the detector's decision.

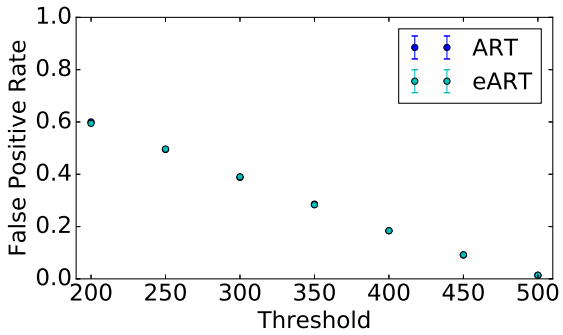
6.2.3 Initial Fusion Approach: Cumulative Fusion

In [141], we aim to demonstrate the advantages of fusion, and do not discuss how node-centric approaches can be incorporated into our framework (we refer interested readers to our earlier work [115] for more details). Therefore, we elected to use the consensus operator to combine two different opinions about a single beacon message, emphasizing that even this relatively simple fusion operation has potential benefits for detection accuracy, when compared to the individual mechanisms. The consensus operator, also called *cumulative fusion operator*, for two opinions about the same event was defined by Jøsang [144]:

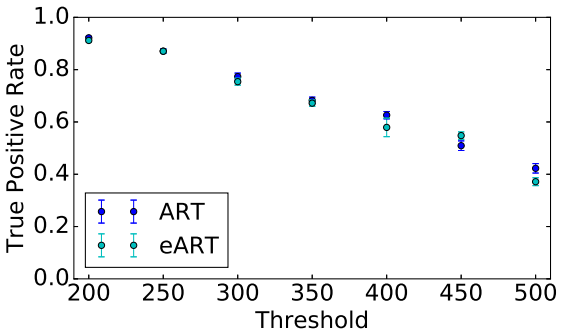
$$\omega_A \oplus \omega_B = (\frac{b_A \cdot u_B + b_B \cdot u_A}{k}, \frac{d_A \cdot u_B + d_B \cdot u_A}{k}, \frac{u_B \cdot u_A}{k})$$

where $k = u_A + u_B - u_A \cdot u_B$. This essentially combines the opinions by weighting the evidence from each actor by the uncertainty of their counterpart. In other words, the uncertainty of A is "filled" by the certainty of B, and vice-versa. The factor k is used to normalize the result.

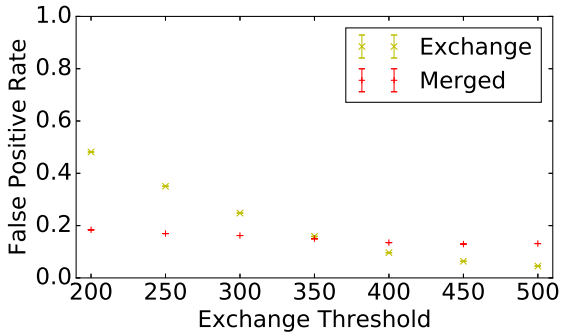
In our implementation of the neighbor exchange, we essentially implement a conservative approach – we assume that which was received previously is accurate. As related work has already shown [30, 33], it is better to consider the age of information, as well as its source, rather than just assume that new information will fit with existing information. However, this essentially represents a node-centric approach to detection, which we considered out of scope for this earlier work – our aim was to show that even a fusion of data-centric mechanisms alone improves detection results.



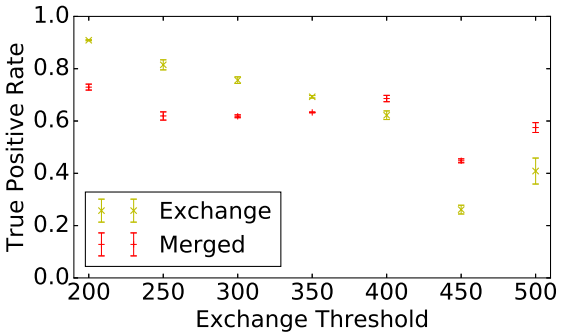
(a) False positives



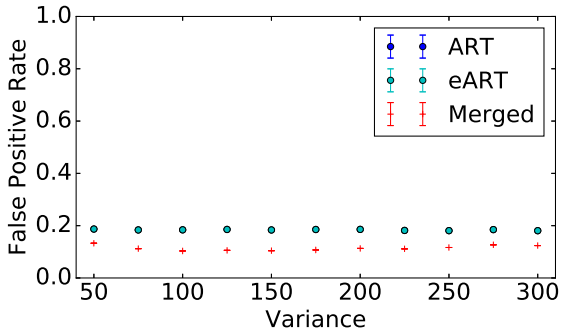
(b) True positives



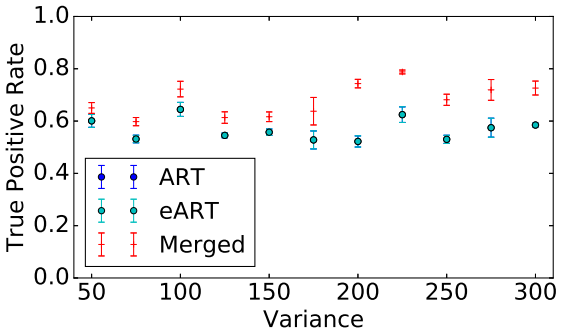
(c) False positives



(d) True positives

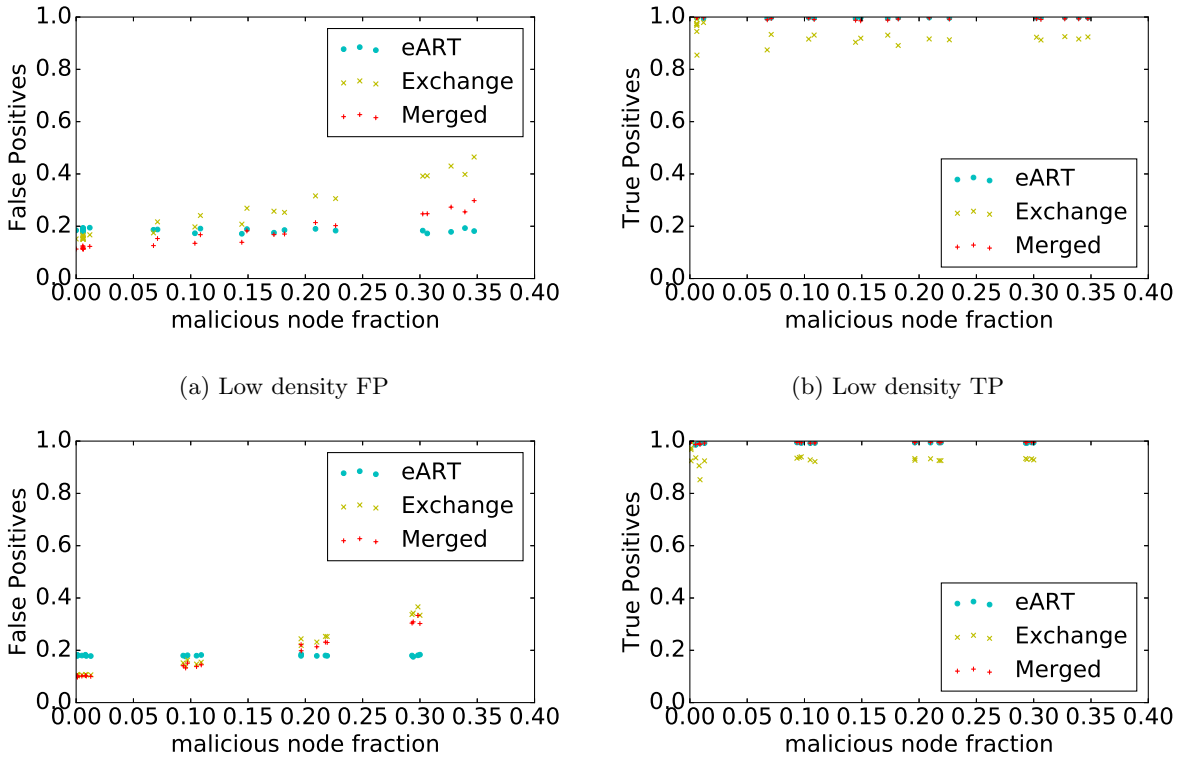


(e) False positives



(f) True positives

Figure 6.6: Comparing different thresholds and variances to configure our enhanced detectors.



(a) Low density FP

(b) Low density TP

(c) High density FP

(d) High density TP

Figure 6.7: True and false positives for low and high density networks against randomized attackers (i.e., transmitting random positions). Low density is after 2 hours of simulation, high density after 6 hours.

6.2.4 Fusion Feasibility Study

Methodology

We use the Veins framework [80] for simulating VANETs, which uses the OMNeT++[5] discrete event simulator to simulate the communication network and SUMO[6] to simulate vehicle movement. SUMO needs map and load data as input – for this purpose, we use the LuST[7] [125] traffic scenario, which is based on real traffic data in the city of Luxembourg. We selected an area in the middle of the Luxembourg map and started our simulation at three different points in time to simulate different traffic load (and therefore different channel loads), as shown in Table 6.1. We implemented our detectors in an application layer class that is integrated into the standard VEINS example. Further simulation parameters are also shown in Table 6.1.

For all of the following graphs, we use *ART* to refer to a fixed threshold, *eART* to refer to our enhancement of the ART mechanism, *Exchange* to refer to the pro-active neighbor exchange, and *Merged* to refer to the fused result. In all graphs, we compute the false positive and false negative rate of different mechanisms. The false positive rate is computed by dividing the amount of detected non-malicious messages by the total amount of received non-malicious messages, while the false negative rate is computed by dividing the amount of attacker messages that were not detected by the amount of received attacker messages. This distinguishes our work from that of Leinmüller et al. [30], who used a non-standard metric. In their work, they use a detection rate and a false positive rate [30]:

> The detection rate is given by the ratio of successfully identified position faking nodes versus the total number of position fakers. In contrast, false positives describe the incorrect accusations.

[5] Version 4.6
[6] Version 0.25.0
[7] Version 1

In other words, the detection rate refers to whether any vehicle in the total simulation ever flags a malicious vehicle as malicious. It is not clear what happens when some vehicles accuse a vehicle correctly, while others do so incorrectly. In our opinion, this metric is poorly chosen, since it causes the results to be inherently dependent on the density and size of the simulation. Since density also affects traffic behavior, which in turn likely affects the performance of this detection (signals may be blocked by obstacles, for example), this seems inadequate. It is desirable to understand the actual effects of density separately from the effects that are caused by the experimental design, and these conflated together here.

Parameter	Value(s)
Channel Models	TwoRayInterference JakesFading LogNormalShadowing
PHY&MAC model	standard VEINS 802.11p
Thermal noise	-110dBm
Bit rate	18Mbps
Carrier frequency	5.890 GHz
Transmit power	10mW
Sensitivity	-89dBm
Beacon rate	1 Hz
Attacker probability	$\{0.01, 0.1, 0.2, 0.3\}$

Table 6.1: Simulation parameters: full source code is available on-line[9]

Parameter configuration

To select the ART threshold, which should approximate the transmission range, we chose the low density scenario (2 hours into the simulation), within which we executed our application on all nodes for 360 seconds with a single attacker (5 repetitions). We chose an easy to detect attacker (which adds the vector $(300, 300)$ to her actual position) for these experiments. This gave us the graphs in Figures 6.6a,6.6b, showing the false positive and false negative rates over the different threshold values. The rates were computed by a weighted average over the simulations. This should essentially reproduces the results from Leinmüller et al. [30] in a more realistic simulation setting, with a moving attacker. However, the false positive and negative ratios are too high for what they assumed is the transmission range (250m). Based on these results, we select 400m as the better choice – trading some detection accuracy for less false positives.

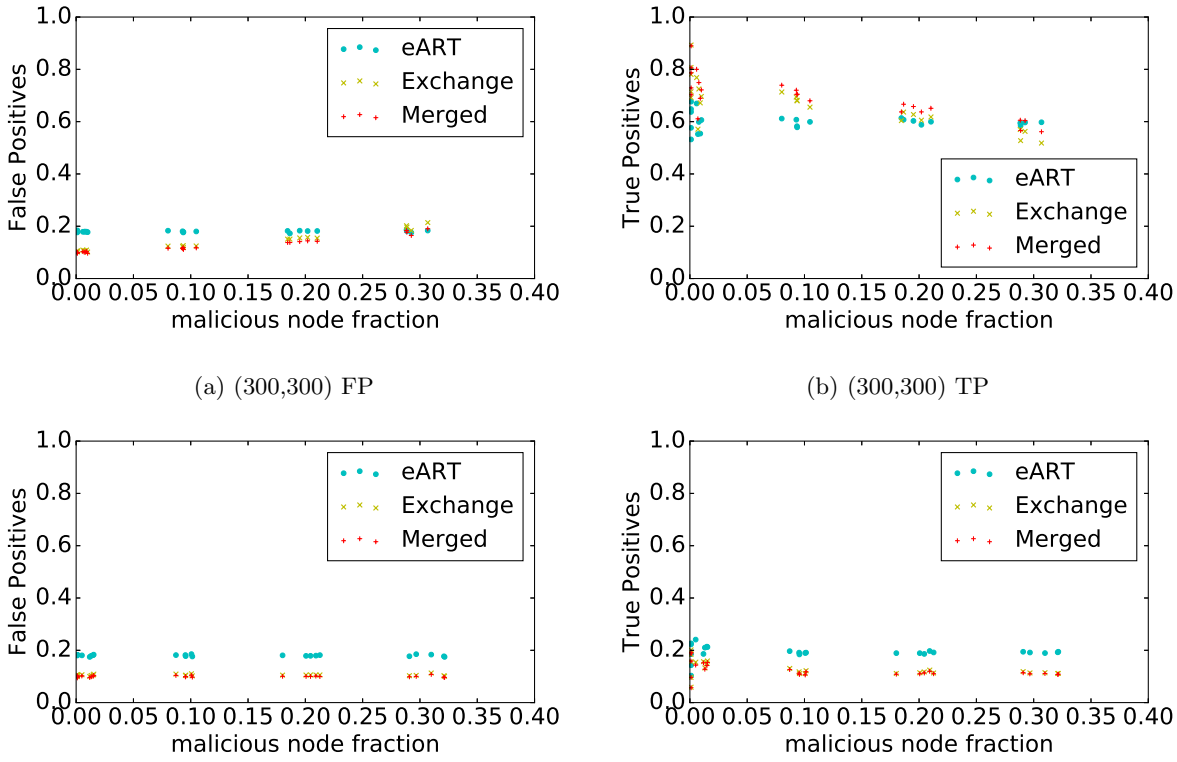
Next, we were interested in whether a different threshold for the pro-active neighbor exchange would make sense. We performed the simulations again, this time varying that threshold, and arrived at the results shown in Figures 6.6c,6.6d. Because the true positive rate drops off at a threshold of 400m[8], we selected the threshold to be 350m. This number also reflects the typical maximum distance between receiver and legitimate sender in our simulations. Having set the essential thresholds of both mechanisms, we looked at the influence of the variance parameter for our enhanced ART. The results of this analysis are shown in Figures 6.6e,6.6f, which shows that the merged results of the individual mechanisms slightly outperforms the basic ART in both false positive and true positive rate. This can be explained by the fact that only messages with significant uncertainty are influenced by the pro-active neighbor exchange, which is exactly what we aimed to achieve.

Results & Discussion

The next step in our evaluation was the analysis of the influence of three further factors: different traffic densities, different attacker types and different fractions of attackers. Compared to earlier work, we assume that our attackers are typical vehicles, i.e., mobile nodes, rather than stationary attackers. We formulate several attack strategies:

- a fixed modification (as above) where a specific vector is added to the nodes' current position
- a randomized modification, where a random position from the actively simulated area is chosen

[8]Later analysis has shown that the increase at 500m in Figure 6.6d is due to the accuracy paradox.



(a) (300,300) FP

(b) (300,300) TP

(c) (50,50) FP

(d) (50,50) TP

Figure 6.8: True and false positives for different extremes of the fixed modification strategy.

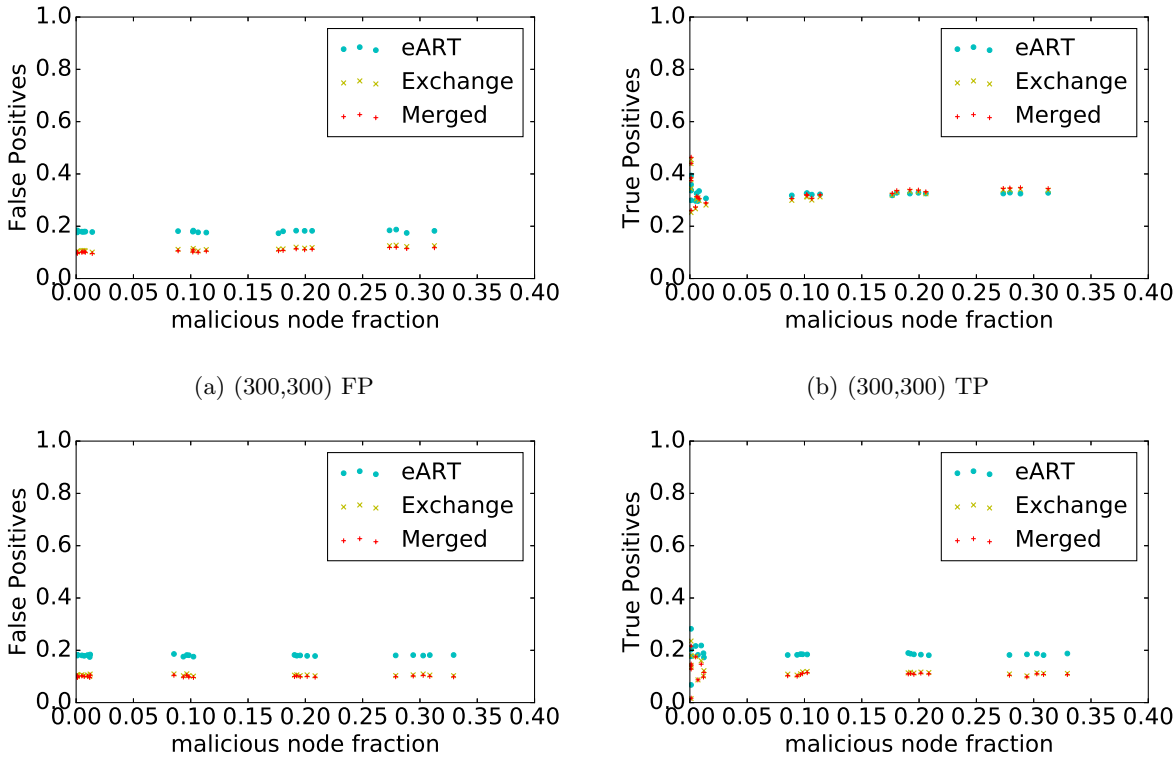
- a randomized vector modification, where a random position in a square around the attacker is selected

To vary the traffic density, we selected three sections of the LuST scenario; the situation after 2, 4 and 6 hours in the area between the SUMO coordinates (2300, 2300) and (6300, 6300). Because each run uses a different random seed, the individual runs have slightly different densities. The attacker probability in Table 6.1 is the probability that any vehicle added to the simulation is an attacker, which attacks with the selected strategy. Because of the same random seed effect, the graphs in this section contain the actual fraction of attackers on the X axis, and each point represents exactly one simulation run. We present the most interesting results of our analysis here, which later motivated us to design Maat; full results can be reproduced using the public source code[9].

Figure 6.7 shows results for the random attacker strategy, which shows the first surprising result: both our fusion and our implementation of the ART detector are nearly invariant to node density, which contradicts with the result in Figure 8b of the work by Leinmüller et al. [30]. We suspect this difference can be attributed to the way the authors aggregate their results, as mentioned previously. We additionally point out that the way in which they generate density differences may also affect the results. In our approach, the simulated area is fixed, but the amount of nodes increases, while in their case, the density is reduced by increasing the network size. This affects traffic flow in ways that are different from reducing the amount of vehicles present, which affects contact times and thus these results. Second, we observe that our detection approach is quite invariant to the fraction of nodes which are attackers. However, this result is expected, because both of the detectors we have implemented are data-centric mechanisms, one of which is completely autonomous. Also, our attackers do not have a collaborative strategy: one could imagine an attack strategy in which multiple attackers transmit false positions, leading to biased results from the pro-active neighbor exchange.

A second notable feature of our results is the relatively low true positive rate for some situations, particularly those where the attacker's strategy is a relatively small modification of his own position,

[9] `https://doi.org/10.5281/zenodo.1232061`



(a) (300,300) FP

(b) (300,300) TP

(c) (50,50) FP

(d) (50,50) TP

Figure 6.9: True and false positives for different extremes of the randomized modification strategy.

as shown in Figures 6.8 and 6.9. As discussed previously, there are some attackers which cannot be detected by looking only at the transmission range, e.g., attackers that introduce a minimal change in their location. We have chosen not to modify our metric for this case, because it is difficult to find a reasonable definition of *attack* if we want to exclude these cases. In particular, our metric also considers correct messages (i.e., messages where the claimed and actual positions of the attacker are the same) as messages that should be detected. This weakness also exists in the work of Leinmüller et al. [30], as can be observed in Figure 8a of their paper. This leads to a very low true positive rate for attackers that make small modifications to their position. In this section, the main result is that merging information from different sources leads to a reasonable result in all cases. These results were the initial bases on which Maat was developed; we discuss the evaluation of the framework, with the detectors presented above, in the following chapters that cover Maat's evaluation.

6.3 Subjective Logic for Detectors

Now that we have discussed the various detectors and how we chose to generate their opinions, we briefly discuss how opinion generation might be done in general. As subjective logic is fundamentally based on evidence theory, it appears intuitively reasonable that designing detectors to set application-specific values for their detection results should lead to better fusion results, as more information is preserved. In this section, we discuss several options how this might be accomplished, based on the designs discussed above, as well as their advantages and disadvantages. After a discussion of how opinions should be generated, we also discuss the fusionability parameter mentioned in Chapter 5. This parameter is essentially a default trust in every detector, and the value to which it is configured determines the weight of individual detectors over the final result, as well as the amount of uncertainty.

6.3.1 Integrating Existing Detectors

Our initial proposal [115] specifically included as a requirement that existing detectors can be integrated into Maat. Other frameworks that use simpler fusion mechanisms have not described in detail how detector integration works, and due to the added complexity generated by subjective logic, it becomes even more important to specifically address this point. Therefore, we here explain how existing detector output can be included into our framework, by specifying how the subjective logic opinions should be generated in these cases. We first focus on detectors that explicitly evaluate the reliability of data items (i.e., messages), which can be divided into two categories: those that make binary decisions, and those that output some kind of trust value. Binary decisions are easiest to integrate through dogmatic opinions that provide full belief $(1, 0, 0, 0.5)$ or full disbelief $(0, 1, 0, 0.5)$. This is also the approach we initially took for the re-implementations discussed previously.

Mechanisms that output a reliability value are slightly more complex; before integration into subjective logic, a normalization step is required, which depends on the domain of the reliability value. For finite reliability domains, this can be done by simply normalizing the reliability value to the $r \in [0, 1]$ domain, after which a dogmatic opinion $(r, 1 - r, 0, 0.5)$ can then be used as the output. Optionally, one can use *uncertainty maximization* on this output, which generates the opinion with the same expectation that has the highest possible uncertainty. This is a procedure described in the subjective logic book [144, Section 3.5.6] (described for multinomial opinions, but recall that binomial opinions are a special case of multinomial opinions where $|\mathbb{X}| = 2$). For infinite domains of reliability values, one could convert the given reliability value into an evidence assignment in the Dirichlet model; the bijective map from Equation 5.1 can then be used to translate the resulting opinion into an opinion.

Note that converted opinions may preserve information output by the detector, but this does not guarantee maximal performance for fusion. In particular, when Maat is used with a broad set of detectors that behave inconsistently, it is well-known in the data fusion world that it is impossible to improve detection results. In other words, generic fusion cannot provide overall higher performance, since it necessarily relies completely on the detectors – this is analogous to the use of the majority fusion discussed above. However, subjective logic enables detectors to more precisely express the certainty in a particular decision; thus, appropriate detector design enables a fusion process that is not hindered by this fundamental limit. In the next section, we discuss a variety of proposals on how to enhance the performance of different detection mechanisms.

6.3.2 Opinion Generation

There are many ways in which opinions can be configured; the most suitable way depends significantly on the semantics of the detector and the data item. As Maat uses a relatively coarse granularity (i.e., message level evaluation), it is advisable to modify detector output beyond the simple integration process described above. In reality, this will be a trade-off between detection performance, cost and associated risk: designing an optimal opinion for every single possible scenario is not realistic. Therefore, we provide additional recommendations that limit the scope of such work, which are based on generalizations of certain detectors.

One special class of detectors is those that are essentially threshold values that represents an approximate maximum. This class includes the ART and SAW detectors discussed in this chapter. For these mechanisms, a potentially reasonable assignment of opinions has an uncertainty of 1 at the threshold, with the belief and disbelief respectively increasing as the value is closer or further away from the threshold. For example, for the ART, the belief increases as the claimed position is closer to the receiver, while the disbelief increases as the claimed position is further away. This represents the fact that the acceptance range is likely to be strongly dependent on environmental conditions.

The configuration of belief and disbelief change is a choice that should be made with the detector's properties in mind. For the individual detector, the actual choice for this is completely irrelevant, since by definition a positive belief or disbelief combined with uncertainty alone always provides the same decision as a plain threshold. However, the advantage is that the amount of uncertainty directly implies confidence, which is used varyingly by different fusion operators. One generic proposal that we make, which was used for the ART and various other detectors we apply in our tests, is to assume a normal distribution and experimentally estimate the behavior of the tested value by the detector. The opinion for this is given by Equation 6.3.2, where δ is the measured value, and the correct behavior is assumed to be normally distributed $\mathcal{N}(\theta, \sigma)$.

$$\omega = \begin{cases} (1 - e^{-\frac{|\delta-\theta|^2}{2\sigma^2}}, 0, e^{-\frac{|\delta-\theta|^2}{2\sigma^2}}) & \text{if } \delta \leq \theta \\ (0, 1 - e^{-\frac{|\delta-\theta|^2}{2\sigma^2}}, e^{-\frac{|\delta-\theta|^2}{2\sigma^2}}) & \text{if } \delta > \theta \end{cases}$$

Note that this process describes a rough estimation, *not* an accurate evidence vector as done in the Dirichlet model. It is in principle also possible to generate opinions by using the standard frequentist interpretation of a value in the domain of the opinion, and counting the frequencies of true and false statements. The problem with this approach is that it requires a rigid formulation of the domain $\mathbb{X}$ that is the same for every detector's view of a data item. In particular, this also requires that every counting step is independent, and that each trial is a measurement of the same variable. This latter condition is the one that is typically violated in our model, because we do not (just) sample the statistical behavior of message propagation through the network (in the case of ART), but also the relative positions within the simulation. An alternative might be to use a much more precise belief assignment with hyper opinions (which at least in principle allow infinite domains of the form $\mathbb{R}^n$), but the storage costs of such opinions make this infeasible. This is because such opinions would have $2k - 1$ degrees of freedom, where k is the set cardinality of the event ($|\mathbb{R}^n|$). This issue may be solved by future research to optimize this naive storage approach, for which we make brief suggestions in Chapter 11. For the remainder of this work, as discussed in Chapter 5, we instead rely on binomial opinions, which necessarily involves simplifications; this is our justification for using the estimate above. We rely on a combination of fusion and validating tests to determine the correctness of the system, as described in the evaluation part.

Alternatives to Normal Distribution

There are a variety of alternatives to the assumption of a normal distribution as a model for how belief and disbelief should change. Although it is likely that alternative distributions of the belief and disbelief mass will provide a more solid foundation for detectors' opinions, we relegated the task of choosing these distributions for future work. This is primarily because appropriate fusion mechanisms are more the focus of this work; however, there are also two technical reasons for this decision. First, there is a severe lack of real-world data even with regards to *valid* behavior for most detectors. Taking again the acceptance range threshold as an example, choosing the appropriate and statistically valid opinion here requires experimental data on how frequent a message reception occurs that is further away than a certain threshold θ. Propagation and shadowing models provide an initial indication of how this system behaves (which we used to arrive at estimates for reasonable values of θ, as discussed earlier in this chapter).

The second reason we did not investigate this in detail is that the tuning of existing detectors is not scientifically significant. Ultimately, tuning detectors is an engineering task that is better suited for industry, where currently unknown factors such as vehicle properties come into play, which cannot be assessed accurately in advance. Making the required assumptions to come to a meaningful result likely implies that the results we arrive at are difficult to transfer to the real world. We have already briefly discussed that transferring the model used for the expression of opinions is difficult to transfer between detectors; this directly connects to the assumptions required for such a setup. We instead focus on fusion of imperfect detectors, which is something that is much easier to transfer to high quality detectors that may be developed for the real world.

Future work could look for specific classes of detectors that could be modeled with more precision, as well as provide a methodology and tools to make the design of high quality detectors easier. One promising path for this process is to look more closely at the relationship between opinions and Dirichlet models. However, there is a deeper question that should also be addressed, related to data granularity in the world model. Recall that data items in our model refer to claimed facts. One key challenge is facts can be multifaceted (e.g., position, heading, speed claims combined with various other facts), and they are sometimes mutually exclusive. In our current model this complexity should be captured by the detector (i.e., each detector provides self-consistent opinion outputs); one consideration was to extend the world model to represent these facts. As there are no implementations of the necessary subjective logic constructs, especially to map between different domains, we kept the complexity in the detector for this thesis. However, we think this might also be a fruitful direction for future work.

6.3.3 Detector Weights: Fusionability

As discussed in Section 5.5, we use a fusionability parameter that decides the default weights of every detector. In subjective logic terms, it is the trust of the system in this particular detector. Because we fuse the different paths between system after trust discounting through our opinion in the detector, the choice of the fusionability parameter influences how fusion will perform. Note that the parameter is therefore also an opinion that should be chosen. We set the fusionability parameter to be an uncertainty-maximized opinion that quantifies the trust in our detector. As described in detail in our evaluation, we experiment with different values for this parameter.

The essential problem this parameter is aiming to solve is that some detectors may perform poorly, either due to poor design or poor parameterization. The fusionability parameter addresses this challenge by bounding the impact any one detector has on the final fused result. This applies to all subjective logic operators, but not to the baseline operators (such as MIN and MAJ), which do not directly use the relative uncertainties of the input opinions.

Poor performance can have various causes, such as an incorrectly configure opinion (i.e., the dogmatic opinions issued by detectors without support for subjective logic), as well as by errors or attacks. Errors here refer to classification errors made by the detector, which can be caused by programmer error or due to the fact that detectors cannot always distinguish attackers from benign vehicles, particularly if those vehicles behave very close to benign. Similarly, legitimate vehicles sometimes have a behavior that could be considered malicious[10]. There is some spectrum between errors and attacks: an attacker can also intentionally craft messages designed for misclassification, for example by manipulating some fields but leaving others untouched. Building detectors that provide dogmatic output or extreme confidence and fusing these opinions directly is therefore a bad idea, unless such a detector covers all possible attacks. However, with a detector that can detect everything, fusion becomes unnecessary (and we have previously argued that such a detector is fundamentally hard to design). Thus, fusionability is expected to increase performance of the fusion process in most cases.

6.4 Conclusion

This chapter introduced the detectors designed and implemented during Maat's development. Some detectors were found to be unsuitable, namely the OPD (which had very poor performance) and the pro-active neighbor exchange (due to its massive communication overhead). We also discussed early results that we used to assess trade-offs between the ART and the pro-active neighbor exchange, which we used in our early experiments to study the feasibility of fusion. We learned how to design opinions in such a way that their opinions can be meaningfully fused, rather than essentially amounting to a simple majority decision or a minimum (i.e., if either detector detects an attack, this is marked as a positive). We used the cumulative fusion operator to merge these results, but as discussed in Chapter 5, we later determined this operation may not be usable for this purpose. After refining our design to the full Maat design that was described in Part II, we can now move to the evaluation of Maat.

[10] In principle, it is possible to design detectors for which this does not happen. However, such detectors are typically close to useless, because they do not detect the vast majority of "interesting" attacks, i.e., those attacks that lead to some response from the victim's application. For example, the ART could be modified to have the maximum range of a DSRC propagation as a threshold (about 1000m), but most interesting attacks concern manipulated messages whose modification is significantly below 1000 meters.

Part III

Evaluation

Chapter 7

Evaluation Methodology for Misbehavior Detection

This chapter describes the overall evaluation methodology applied in this thesis. We briefly describe the different approaches available to us (simulative, analytical and experimental), before explaining the attacker model and the corresponding attacks. Furthermore, we also consider the available metrics in this chapter, introducing several new ones as part of our contribution. Part of our contribution is to make a significant step forward in the methodological validity of evaluation done in this field, which is discussed throughout this entire third part of the thesis. As an additional component, we discuss an extensive analysis of attacks on CACC, one of the major application areas of intelligent transport systems, previously published at VNC 2017 [159]. The text in this chapter is also partially based on earlier publications at the Fachgespräch Inter-vehicle Communication 2017 [158] and SecureComm 2018 [172].

7.1 Overall Methodology

Cooperative intelligent transport systems are one area where misbehavior detection provides a significant benefit, because integrity is more important than confidentiality, and the data correctness plays a much more significant role than for regular networks. The literature has already developed many schemes, which are evaluated in widely different ways and with very different tools. Some are analytical, some are simulative and sometimes there are limited experimental cases.

Similarly, the data that is verified is usually application-specific, and not independent of other surrounding vehicles (i.e., a ground truth cannot be established for just the particular message content that is being evaluated), which makes analytical approaches very difficult, since one cannot extrapolate results from small examples. Additionally, analytical approaches often require simplifications that could again be exploited by an attacker. Experimental approaches have different issues; experiments are expensive to perform, and cannot necessarily be applied to all settings, especially when the consequences are dangerous (e.g., attempting to trigger a crash on a highway for experimentation is not ethically or financially feasible). Nevertheless, experimental approaches can still provide some benefit in estimating the behavior of a system in an attacker-free environment, and the resulting dataset can be artificially extended with attacks [98].

Work on intrusion detection system (IDS) validation has already seen that it is difficult to compare results between IDSs, but this is made harder by the fact that misbehavior detection systems (MDSs) analyze application-specific data. This data is dependent on real-world behavior and application state, and is therefore more unpredictable than the regular, well-defined behavior of, e.g., network protocols. Many other questions play a role: should detection happen on a per-packet basis or a per-node basis? Is the goal to eventually detect all attackers, or to detect the most attackers with the lowest latency? What kind of attacks are considered? Is there application logic that tries to be fault tolerant?

The evaluation of misbehavior detection mechanisms is often strongly influenced by what kind of attacks the authors were considering. This makes comparisons across published results very difficult. There seem to be no satisfactory evaluation metrics in the literature that both catch the subtleties of misbehavior detection, while still remaining general enough to be applicable to more than just a specific

mechanism.

7.1.1 Evaluation Strategy

Simulation-based evaluation strategies enable reliable and reproducible experiments in a variety of settings, and enable analysis of detector performance on a large scale. Although there has been work on empirical analysis of attacks [98], such studies are extremely difficult to generalize and verify without a significant deployment of vehicular communication systems, which are still in their deployment phases at the time of writing. There is a wide variety of network and vehicle behavior simulation tools, with different scenarios; however, previous authors have already given extensive analyses of such tools [81, 138].

Instead, the question that should be central is what variables could have an impact on overall detection performance, and which should be analyzed individually to make general statements about the suitability of a detection mechanism. Important parameters here include the number of attackers, the number of legitimate vehicles and the driving scenario. By studying these parameters, authors can gain an insight into whether their detector has weaknesses in specific scenarios, and how it performs when significant message loss is encountered, and these are often also used in the evaluation of vehicular applications. In all cases, repetitions under different simulation seeds are used whenever probabilistic models of, e.g., the communication channel are considered. The number of attackers is a first step to determine whether the detector is resilient against attacks.

Resilience against attacks is a metric that is difficult to quantify, however; it is particularly challenging to consider sufficiently distinct attacks, and to implement these attacks correctly. Because detectors are often designed to detect specific attacks, it is tempting to implement exactly these attacks and then show that the detector performs well. Therefore, it is important to clearly specify the attacker model, including how exactly this attack works, and whether or not attackers can cooperate (collaborative attackers) and whether or not attackers can create a limited amount of additional identities (Sybil attackers). In the latter case, the attacker uses different pseudonymous identities that are possibly linkable by an authority; in the former case, multiple independent vehicles are attacker-controlled (e.g., this could be due to malware).

In this thesis, we apply a simulation-based evaluation strategy with a strong focus on reproducibility and extensibility. The goal is, in essence, to make it easy for future research to falsify the results through improved attacks and/or detectors. To this end, we publish all associated code and data for re-use by other researchers. We use the well-known VEINS [80] simulator as a basis for all our experiments, as this is an established open-source simulator that already provides a solid base line for simulation of vehicular networks. In particular, there is a realistic traffic scenario called LuST [125], as well as a platooning extension Plexe [122], which together cover the use cases we are interested in. For platooning, we design an attack that is dedicated to a specific attack goal (disrupting the platoon), which enables us to compare the resilience of controllers. For a comparative analysis of detection mechanisms, we are primarily interested in the ability to detect attacks in the abstract; for perfect performance we require that all messages from the attacker be classified as malicious by all vehicles. This provides a pessimistic estimate of detector performance, which is useful if the goal of the attack is unknown. The attacks we implement are based on state of the art work, and we allow for a high attacker penetration (up to 30%, close to the limit of what is decidable). As we assume no particular attack goal, we do not implement cooperative attacks in this work; the only cooperative effects we will see is that multiple attackers apply the same attack algorithm. This mimics a malware that causes misbehavior within a broad set of vehicles, without requiring communication between the attacking vehicles.

7.1.2 Evaluation Metrics

Having considered the evaluation strategies, we next consider appropriate metrics for validation. Choosing multiple metrics is important, because usually detectors represent some kind of trade-off between different outcomes. It is therefore important to clearly state what the evaluation metric is. This means that different metrics need to be considered, requiring an evaluation strategy and study design that is as independent from the evaluated mechanisms as possible.

Most papers discussing detectors evaluate these through some measure of detection quality. However, there are many different strategies and nuances to this process, both in terms of the chosen metrics and

Table 7.1: Example confusion matrix for a detector. The detector classifies each message as malicious or legitimate (columns), and the base line is represented by whether it is an attacker or benign message (rows). The items in this matrix correspond to true positives (100), false negatives (50), false positives (150), and true negatives (9700).

	Classified as Malicious	Classified as Legitimate
Attacker Message	100	50
Benign Message	150	9700

how the results are analyzed. Because misbehavior detection specifically is something that is fundamentally dependent on application data, generalizing results is more difficult than for regular classification tasks. This is particularly relevant when evaluating the large-scale effects of wide-spread misbehavior. For example, consider position falsification attacks: the ability of detectors to correctly classify this attack greatly depends on the receiver's position relative to the sender. As such, it matters how the detection results are over different vehicles. Similarly, it is important to consider whether one is evaluating immediate detection (i.e., directly after reception) or eventual detection (i.e., does the receiver eventually recognize the attack). In our work, we focus primarily on the former, and in that case the question is also raised whether we should compute our metrics per vehicle, and then computing statistics over the vehicles, or whether we should aggregate all reception events and applying the metric to the result.

Confusion Matrix

The most obvious approach to evaluate the quality of a detector's results is to use the confusion matrix, which is also used in other fields (e.g., medicine and machine learning). This matrix essentially counts false positives, false negatives, true positives, and true negatives, as well as their rates with respect to the entire population. An example of such a matrix can be found in Table 7.1. Metrics derived from this include accuracy, which is the amount of correctly classified events divided by the total population, and false omission rate, which is the amount of false negatives in the set of all negatives. Most papers in the area of misbehavior detection that use this type of metric use the false negative rate to quantify the risk of missing detections, and the false positive rate as the risk of an incorrect detection event. Opinions differ widely on what acceptable values are (e.g., for intrusion detection in networks with a lot of traffic, a false positive rate over 0.001 per packet is considered very bad), and it is difficult to make general statements about this, because the impact of a false detection event is significant.

However, it is actually not trivial to classify a detectors' results into these categories when evaluating the detector in a simulation. Recalling the previous example, in distributed detection scenarios such as C-ITSs, proximity to the attacker plays a significant role in how likely a vehicle is able to detect an attack. Not all nodes in the network will hear all messages, so if the data is aggregated across different detectors, one must take care to normalize these results. Instances of a detector do not necessarily produce the same output for a given message (due to differences in internal state between vehicles), and not every message is seen equally often, due to the broadcast nature of these networks. Simple normalization may not be sufficient here: if a specific subset of receivers produces a very high false negative rate, while the aggregate of all receivers on average performs very well, this could still mean the detector is bad. In other words, aggregating and normalizing over an entire simulation may hide the fact that there is a weakness in a specific scenario.

Having established a way to normalize and aggregate the detection results still leaves other questions open. The definition of the input of the detector is one of these factors: does the detector take a single message and output a classification, or does it take a stream of messages and output an eventual classification? In the latter setting, one needs additional metrics to establish the timeliness of the system (e.g., detection latency: how long does it take for a detector to correctly classify an attack, after this attack starts?).

For a data-centric setting, using the confusion matrix is particularly difficult, because of the fact that many attacks are impossible to distinguish from legitimate messages. This can happen in two cases: either the message was sent by an attacker, but follows the expected behavior of vehicles (i.e., it is not a malicious message), or the attack is so marginal that it has no impact (and is thus indistinguishable from expected behavior or sensor noise). For example, an attacker might transmit a beacon with a position a few centimeters from its actual position. This led some authors to conclude that application-oriented

evaluations may be more suitable: if the attacker cannot achieve a goal, because the impact of false data is too small, then clearly the detection mechanism is effective. The disadvantage of this strategy is that the evaluation depends not only on the simulation aspects and the attackers' implementation, but also on the application implementation.

For data-centric detection mechanisms that are based on consistency, i.e., they consider multiple data sources and detect inconsistencies, it is often implicitly assumed that previously received data is true, and only the incoming packet is classified as legitimate or malicious. However, this means that message order is particularly significant: whenever a malicious message arrives first, it may trigger an additional false positive, because the next legitimate message differs from the malicious message.

Many papers implicitly discuss that detectors should also perform revocation or response – they exclude specific messages from those that are received, in order to prevent errors in the application. Similarly, some detection algorithms are inherently incompatible with the idea of evaluating individual message on a sequential basis, because they perform some batch processing (e.g., classifying vehicles instead of messages [71]).

Some authors use a pre-classified set of messages, which is not necessarily data-centric (e.g., Grover et al. [71] use a set of 3101 legitimate and 1427 malicious samples, with several types of attacks). This is common in the field of machine learning, where classifiers are often tested using this type of approach. However, since most detection algorithms in C-ITSs have different inputs, it is difficult to find a conclusive set that considers these various inputs, as well as contain the necessarily distinct types of attacks (especially when reputation is considered, which can be built over time). In other words, this approach is only valid if each sample is well-defined, which is not the case in our more general setting. Finally, note that these message samples are not publicly available (as far as we are aware, the only exception here is our work [172]).

Alternative Accuracy Metrics

This led authors to use application-specific or detector-specific evaluation strategies, in order to demonstrate specific strengths of individual detectors in comparison to others from the literature.

Application behavior metrics In many C-ITS scenarios, the specific values transmitted by an attacker are not necessarily relevant, but what is of interest is whether a receiving vehicle makes incorrect decisions about the state of the world. Application metrics aim to capture this subtle concept into a concrete metric. These metrics are useful, because they can also consider errors from other (i.e., non-malicious) sources, and are independent of a deep understanding of the detection mechanism. However, they require an application implementation, which is bug-prone and makes attacker implementation more complex.

One specific category application metrics is that related to schemes that detect routing misbehavior. Because routing misbehavior is historically closely linked to evaluation of routing schemes, some authors use routing performance metrics (such as arrival rates, consumed bandwidth and similar metrics) and changes to vehicle mobility [30, 51].

Another class of application metrics is much closer to the data that many data-centric detection mechanisms analyze; for example, this includes collision avoidance applications [53, 75] and in-network aggregation [103]. A disadvantage of these strategies is that it is hard to use them as a baseline for other studies, because often they are very specific.

Detector specific metrics Some detectors have known sources of potential errors, often inherent to their design; a common strategy to deal with this is to approach and analyze these issues specifically. This is particularly useful when it is very clear what kind of error sources are to be considered. For example, Bißmeyer et al. [45] used this type of metric to also include GPS error as a potential source of additional false positives for their scheme to analyze these effects in detail.

Another example of such an approach would be an evaluation the impact of the attacker on the analyzed variable. This only works for continuous variables, such as position information, where a simple distance metric is available to estimate the error between real positions and falsified positions. Rather than looking at how well attacks are detected only, this strategy would measure the distance between accepted attacker-generated data that is accepted as valid, and use this as a metric for detection quality. This approach is also viable if error sources are considered for legitimate vehicles. A disadvantage of this

approach is that it may only be suitable for specific classes of detectors (or at least, it may put other detectors at a disadvantage).

Other Metrics

There are many other types of metrics available to authors seeking to evaluate their detection mechanisms. Some notable examples include:

Detection Latency The time required for an attack to be detected. The exact definition of this metric differs type of detector, but typically it is the time between the first malicious packet received and the first detected malicious packet. This is particularly significant to measure the impact of reputation abuse in trust-based node-centric detection mechanisms: if trustworthy vehicles transmit malicious information, the potential impact of an attack is large.

Computational Cost Although relatively uncommon for misbehavior detection in vehicular networks[1], more traditional benchmarks such as computational cost can also be used. Most authors only check whether reasonable estimates of locally received messages can be analyzed in reasonable time (which, using 100 vehicles in range transmitting at 10Hz, is at most a millisecond per message), but particularly with proposals that include multiple detectors, scalability could be an issue.

Financial Cost Traditional IDSs are commonly evaluated by examining the cost associated with response and the corresponding variants in the confusion matrix [15]. In the case of vehicular networks, this is a bit more complex: the cost of a false negative is difficult to estimate, and strongly dependent on the scenario (i.e., high speed collisions have much higher cost, even though the classification of the event is exactly the same). Estimating this cost is also ethically complicated, because human lives are included in this process, which suggests that such an evaluation requires more extensive knowledge on how to deal with this suitable (as done, e.g., in invasive medicine).

Stability & Usability One factor that is often forgotten or stated as future work only is that the detection mechanism should be sufficiently stable, such that the users' experience with the system is good enough. This is important, because users' trust in the system is strongly dependent on their perception of the systems' reliability. If the system continually warns or makes changes in response to possible attacks (high false positives), but has an overall high accuracy than other systems, then users may still perceive that system as very unreliable.

7.1.3 Recommendations for Simulation-based Evaluation of MDSs

We recommend authors to follow the developments in the simulation community, and make use of extensive standardized scenarios, such as the LuST scenario [125] for traffic simulation. This also includes pro-actively porting source code that is under development to more recent versions of the simulation environment wherever possible: ideally code should always be based on the most recent stable version of the simulator available while the paper is under review. This allows authors to benefit from improvements in the simulation environment and simultaneously consider new features, such as newer channel models.

For reasons outlined above, we also recommend using a variety of different attack strategies, which ideally have very different goals. This ensures that the authors can describe the qualities, as well as the limitations of their detector, which in turn can help future authors decide whether new attacks against this detector may be possible. It is also important to define a baseline in addition to these attacks, which can be undertaken in various ways (and this depends strongly on what the authors actually designed), and if possible a simulation of potential non-malicious sources of error. The simplest example is GPS error: adding an error to a GPS coordinate is relatively simple, but it makes the results much more representative than an idealized perfect positioning system for each vehicle, where it just "knows" its position.

The authors should at least consider how they aggregate the detection results across messages, vehicles and simulation repetitions, as discussed above, and clearly specify their approach. If possible, we

[1] We are not aware of any such studies.

recommend developing the simulation in such a way that multiple aggregation approaches can be used; there may not be a one size fits all solution.

Finally, we recommend publishing source code[2], or at least making it available to other researchers on request. This enables reproducibility, one of the core principles of scientific research, which allows research in this area to make faster and more meaningful progress. It also enables studies that analyze the behavior of a variety of algorithms in different settings a much more efficient and less error-prone process. Finally, the authors themselves benefit from this process, because it is easier to compare to the literature.

7.2 Metrics for Misbehavior Detection

In this section, we discuss the different metrics that are used in the evaluation of Maat and C-ITSs in general. We identify three categories of metrics that can be applied effectively: large-scale detection performance metrics, detector limitations and application metrics.

7.2.1 Background

Detection performance is a complicated and multi-faceted issue, whose definition also varies across publications, depending mostly on the purpose of the detector. Even in intrusion detection in general, it is considered a challenging, non-trivial trade-off to determining how to evaluate detection mechanisms and how to choose the appropriate mechanism for deployment [15]. In misbehavior detection specifically, many authors use false positive/negative rates or equivalent metrics to determine how well attacks are detected, and this is combined with other performance metrics (such as latency, or application-specific metrics). Although these metrics are useful to compare performance of mechanisms, we find that there is a lack of metrics that are useful to figure out where weaknesses of existing detectors are. In this section we propose an additional metric that fills this gap, which can be used by developers to tune detectors, and to determine where new detectors are needed.

Another issue that should be addressed is specific to detection in *distributed* peer-to-peer systems: how should detection metrics be aggregated across participants? For example, given a simulation with two honest vehicles and one attacker, how do we characterize the detection performance of the same detection system (running within two vehicles independently)? We previously touched on this issue in a discussion with the vehicular communication community [158]. In our evaluation, we will quantify the detection quality by classifying every detection event as true/false positive or as true/false negative; a detection event occurs whenever a message is received (i.e., we assume the detection decision is made as soon as possible after reception). We aggregate these results by counting the errors generated by *detection events*, not in terms of *sent messages* or *participants*. Which aggregation method is chosen is highly relevant for the interpretation of the results: in this work we focus on detection events to obtain a picture of the overall quality of the results. Aggregating by *message* provides information about how well a specific message sent by the attacker is detected, but presenting the results in terms of detected messages would mean that the amount of receivers is completely disregarded. Similarly, aggregation by *participant* ignores how much contact this vehicle has with the other vehicles. Since the amount of messages between vehicles is also indicative of a potential impact of an attack, aggregation by detection events is the best approach for an overall evaluation of detector performance. However, we point out that these metrics can also be implemented with our dataset, since we provide message and sender labels for every message.

7.2.2 Large-scale C-ITS Metrics

The first metric we use to decide the quality of the detector is based on the confusion matrix, as mentioned previously. An intuitive choice for large-scale evaluation might be *accuracy*, which defined as the number

[2]This goes back to a common argument in computer security with respect to responsible disclosure. A commonly raised concern is that this process will enable attackers to compromise vulnerable systems. In this specific case, there is an easy refutation to this argument: first, attacks on simulations are not (necessarily) attacks on real systems, and second, a scientific publication should already contain sufficient details for reproducibility. In both cases, implementation effort is the only thing that separates cases where source code is not public from those where it is. This effort is considered negligible; however, publishing the source code makes it much easier to build detection systems, improving the overall security of many systems.

Table 7.2: An evaluation of an imbalanced dataset may lead to misleading conclusions if the accuracy is used to decide the optimal classifier. In this example, there are 10000 messages to be classified, of which only 100 are malicious. Therefore, a classifier that attempts to correctly classify everything (**MDS 1**) performs significantly worse on this dataset than a classier that simply assumes there are no attacks (**MDS 2**). This is referred to as the accuracy paradox.

MDS 1

	Classified as Malicious	Classified as Legitimate
Attacker Message	100	50
Benign Message	150	9700

Accuracy of **MDS 1**: $\frac{9700+100}{9700+150+50+100} = 98\%$

MDS 2

	Classified as Malicious	Classified as Legitimate
Attacker Message	0	150
Benign Message	0	9850

Accuracy of **MDS 2**: $\frac{9850+0}{9850+0+150+0} = 98.5\%$

of correct classifications ($TP + TN$) over all classifications ($TP + FP + TN + FN$), appears intuitive but suffers from the accuracy paradox for imbalanced sets. This is illustrated in Table 7.2. It is thus considered good practice [16, 137] to always provide a quantification with two values, showing the trade-off between increased false positives to reduce false negatives and increased false negatives to reduce false positives. One such formulation is the use of *precision* and *recall*: precision quantifies the relevance of detection events ($TP/(TP + FP)$), while recall quantifies what rate of positives is actually detected ($TP/(TP+FN)$). An optimal detector thus has a precision and a recall of 1; how significant a deviation from this value is acceptable depends strongly on the application.

The state of the art [170] typically reports false positive ($FP/(FP+TN)$) and true positive ($TP/(FN+TP)$) rates, which provides a different and significantly skewed picture in certain situations, as discussed in machine learning literature [16, 137]. Specifically, precision and recall are more informative in situations where a binary classification task (e.g., packet maliciousness decisions) is performed on an imbalanced dataset. As our dataset contains attackers in different degrees, and the amount of decisions made for attacker-transmitted messages ($TP + FN$) compared to the amount of decisions made for benign messages ($FP + TN$) is significantly different, we should thus prefer precision recall curves. As pointed out by other authors [16], a detector that is better in the precision-recall (PR) graph is guaranteed to be better in the receiver-operator curve (ROC) graph; the interpretation process is generally similar (i.e., which curve is closer to the optimal point).

PR graphs provide us with an overall estimation of detector performance, but they have an important disadvantage: they are generally not as easy to interpret as a graph with FPR/TPR (referred to as an ROC curve). This greatly impacts the use of PR graphs in the literature: not only are they somewhat harder to understand fully, PR graphs often "look much worse", as demonstrated by Davis & Goadrich [16, Fig. 1]. This figure shows that the ROC curve can look close to optimal (the *area under curve* (AUC) is large), while the PR curve for the same data looks much worse (the AUC is small). This is partially related to the fact that interpolation between points on a PR curve is non-trivial; for details, refer to [16]. In addition to this issue, PR-graphs do not provide information about where potential flaws of individual mechanisms are, or whether a combination of multiple detectors can out-perform the individual mechanisms (as we argue in previous work [115]). In this work, we this use the PR-graphs in combination with our new metric, based on the Gini index.

7.2.3 Evaluating Detector Limitations

To study the limitations of detectors without arbitrarily guessing which factors may influence such detectors, we design a new metric to find indications of such influences. The idea of our metric is quite simple: examine whether the distribution of erroneous classification rates (i.e., false positives and false negatives) is uniform over the receiving vehicles. If this metric says the distribution is uniform, the detector performance is not dependent on factors that are varied in the simulation, such as which vehicle

executes it, or the relative position between the receiving and sending vehicle. On the other hand, if this distribution is extremely skewed, the conclusion is that the detector performance depends a lot on the context of the vehicle. We expect this is the case for many misbehavior detection schemes (and indeed most of the literature just assumes this is true), but it is also valuable information to know where the discrepancies occur. This enables further investigation into the detector's strengths and weaknesses, and finding a skewed distribution would suggest that combining the results of different mechanisms is the way forward. Note that this is *not a qualitative metric*: uniformly good or poor error dispersion does not imply that a metric is significantly better or worse, it only suggests whether there is room for improvement.

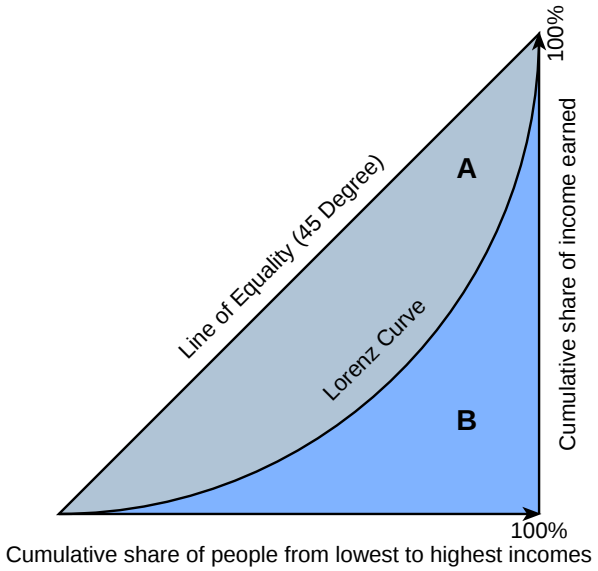


Figure 7.1: This figure shows an example of the Gini coefficient as used in economics. The Gini coefficient is computed as the ratio of the marked areas around the Lorenz curve: $A/(A+B)$. The Lorenz curve represents the cumulative plot of the population income after sorting from low to high income.

Given this intuition, we investigated and found a metric for statistical dispersion that is commonly used in sociology and economics to measure income inequality: the Gini coefficient or Gini index [177]. The idea can be visualized by sorting people by income in ascending order and then plotting the cumulative fraction of this list against the cumulative income of that group. This can be seen in Figure 7.1[3], where the Gini coefficient is used to quantify income in-equality. The coefficient is computed as the ratio $A/(A+B)$; as the Lorenz curve approaches complete equality, the area A approaches zero; a high Gini coefficient thus corresponds to high inequality. More formally, the Gini index G of a population with mean size μ and value x_i assigned to individual i is defined as [177]:

$$G = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} |x_i - x_j|}{2n^2\mu} \tag{7.1}$$

The Gini index itself is not novel, nor is its application to quantify errors (see e.g. [153]), but our application of it is slightly different: we propose that it can be used effectively to determine the statistical dispersion in the error rates across vehicles. The reasoning is that computing the overall performance as discussed in the previous section hides localized effects associated with individual vehicles. Thus, if a mechanism has some regions where it performs really well (e.g., a highway), while it performs very poorly in other regions (e.g., urban settings with lots of traffic lights), these effects will be averaged out in the overall performance. If the overall performance is reasonable, one can use the dispersion in the error rates to determine whether this happens both for false positives and for false negatives (the latter being dependent on the attack): the higher the Gini index of these rates, the more differences exist between

[3]This is a public domain figure, taken from: `https://commons.wikimedia.org/wiki/File:Economics_Gini_coefficient2.svg`

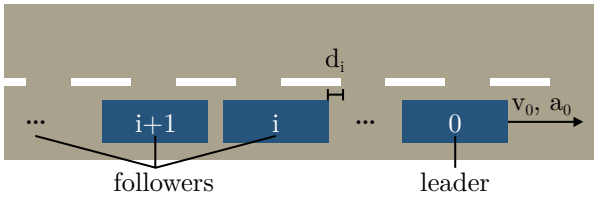


Figure 7.2: An example of a platoon, with the leader in front and a list of followers behind it. The headway distance d_i refers to the distance that the ith vehicle observes to the vehicle in front of it.

vehicles. However, if performance is poor overall, the Gini index can still be close to zero (or conversely, be close to 1); the arrays $(0.1, 0.1, 0.1, 0.1)$ and $(0.9, 0.9, 0.9, 0.9)$ have the same Gini index of zero. There are at least two main ways to use the result of this metric: 1) investigate the vehicles on either side of the skew and see whether the detector can be improved by changing its functionality or 2) investigate whether fusion can be used to exploit low amounts of errors produced by different detectors in different scenarios.

7.2.4 Application Metrics: CACC

For our work on CACC, we aim to analyze the effectiveness of various attacks on different CACC controllers. Recall from Chapter 2 that CACC is an application in which a group of vehicles, referred to as a platoon, aims to increase traffic efficiency by reducing the following distance between vehicles. Essentially, the idea is that communication allows faster response, and thus a lower following distance. An example of a platoon is shown in Figure 7.2. The coordination of a platoon is typically performed by the leader, who is responsible for managing the platoon [121] (e.g., vehicles joining the platoon). Each vehicle employs a controller that takes as input the measurements and received information, producing as output a new desired speed for its vehicle. Different controllers use different information sources; some use all messages exchanged in the platoon, while others only use the leader or the preceding vehicle (i.e., the vehicle directly ahead).

The goal of our attacker is to either cause a crash or maximize instability of the platoon. As such, we distinguish between two cases: a crash or some instability. If a crash occurs, impact velocity Δv is used to quantify the impact of the attack (medical research suggests that this is the best predictive factor for injury [5]). If there is no crash, we quantify the instability in the platoon for the remainder of the simulation (approximately 30 seconds). Since all the studied controllers have been shown to be string stable, we can assume that the platoon eventually stabilizes. The option we chose is $max_{s,i,t}(e)$, where s is the simulation run, i is the ith vehicle in the platoon, t is time and e is spacing error, which most accurately represent the worst case. This was inspired by earlier work [150], where a similar metric is used. The spacing error is defined as the difference between the desired distance of the controller and the actual distance between two vehicles. The alternative metrics were $avg_s(avg_i(max_t(e)))$, the average maximum spacing error, and $avg_s(avg_i(max_t(a)))$, the average maximum acceleration, both of which can be considered metrics for passenger comfort. Usability studies have shown [55] that spacing is important for user acceptance; we included the acceleration as an additional metric, which is also common in transportation research. Averaging over simulations and vehicles allow us to estimate the average comfort of all vehicles in a platoon; however, the maximum spacing error more accurately represents the potential inefficiency and risk due to an attack, which is why we decided on this metric[4].

7.3 Studied Attacker Models

For the metrics discussed above, attacks are needed to determine how well a system performs. Our attacker model is consistent with that discussed extensively in our survey article [170], an improved version of which is also discussed in Chapter 1. However, we constrain ourselves to attacks that are misbehavior, rather than also including a variety of cryptographic attacks. This is in part because existing work already adequately defends against cryptographic attacks (e.g., the US SCMS [168]), and

[4]The data repository for the paper contains data necessary to deduce the other metrics; see `https://github.com/vs-uulm/vnc2017-CACC-data`.

Table 7.3: Attacker parameters for large-scale attacks

ID	Attack	Parameters
1	Constant	$x = 5560, y = 5820$
2	Constant offset	$\Delta x = 250, \Delta y = -150$
4	Random	uniformly random in playground
8	Random offset	$\Delta x, \Delta y$ uniformly random from $[-300, 300]$
16	Eventual stop	stop probability $+= 0.025$ each position update $(10Hz)$

for our work we aim to detect attacks that cannot be protected against using existing means. Similarly, we exclude the parallel field of routing attacks from our scope, because this field is nearly entirely orthogonal to the attacks we consider. The attacks we consider instead focus on targeting safety and driving comfort applications. In the broadest sense, this means that message and sensor manipulation attacks are those of primary interest to our work.

For the papers written in the context of this thesis, the attacker model has evolved slightly over time, and thus the attacks we include are distinct for each work. For example, in the specific case of platooning we also consider jamming attacks, as in the time-critical CACC application, such attacks are particularly effective. Similarly, for large-scale analysis we require different attack types: these attacks are discussed in the remainder of this section. We specifically point out that for large-scale analysis, we did not choose the best possible attacks, but rather a broad set of attacks that is deployed independently in a significant fraction of vehicles. In the real world, this scenario is much more likely than a targeted attack on a specific vehicle: a vulnerability for one manufacturer likely affects a significant part of all vehicles. We expect that the exploitation of such a vulnerability would lead, in the worst case, to an attack that aims to cause wide-spread disruption through relatively simple principles. This is the attack class on top of which we build the VeReMi dataset, which will be described in detail in Section 8.1; here we focus on the attacks specifically.

7.3.1 Attacker Model for large-scale C-ITS

We implemented a set of attacks associated with position falsification, the type of attack that is most well-studied in this field, and for which many mechanisms have been designed [170]. Rather than implement a broad set of complex attacks, we focused on this specific attack to show the efficacy of our approach. The attackers we implement within this class, also shown in 7.3, are the constant attacker, the constant offset attacker, the random attacker, the random offset attacker, and the eventual stop attacker. The constant attacker transmits a fixed, pre-configured position; the constant offset attacker transmits a fixed, pre-configured offset added to their actual position; the random attacker sends a random position from the simulation area; the random offset attacker sends a random position in a preconfigured rectangle around the vehicle; the eventual stop attacker behaves normally for some time, and then attacks by transmitting the current position repeatedly (i.e., as if it had stopped). The random attacks (4 and 8) take a new random sample for every message. The parameters for our attacks are shown in Table 7.3; the numbers are based on previous work [141]. The corresponding attack implementations are used to generate a dataset, which is discussed in more detail in Chapter 8.

7.3.2 Attacker Model for CACC

In this section, we describe our attacker model, which covers attacks proposed in previous work [123, 127, 150], as well as elements from the area of misbehavior detection [141] and attacks on sensors [135]. In particular, we are interested in how realistic the attacks proposed in the literature are, when considering currently standardized security mechanisms. Standardized security mechanisms include the use of pseudonymous certificates to protect driver privacy, the corresponding private keys of which are stored in a hardware security module (HSM), which is somewhat protected against tampering. The core of our attacker model is that an attacker may gain control over a vehicle, and thus over legitimate key material. This can happen either through software compromise (e.g., malware spread through applications running on the entertainment system), by physical manipulation of the vehicle, or through key extraction from an old HSM. However, unlike traditional network attacker models, the attacker additionally must obey

the laws of physics that are observable by any network participant (e.g., an attacker is not omnipresent throughout the network).

For CACC specifically, we assume the software and protocol implementations are suitable, i.e., the attacker cannot find an exploit here, and must obey the protocol. In other words, we restrict the target of the attack to the platooning application, and assume other mechanisms are in place to filter out invalid or improperly formatted messages. In our model, the attackers' primary goal is to cause a crash within the platoon, with the secondary goal of destabilizing it. We assume the attacker is already part of the platoon, a position from which the attacks can do the most damage. However, as the literature points out [127], if the attacker controls the leader vehicle, they have complete control over the platoon, and therefore attacks should be performed by a non-leader vehicle. We follow this argument and also assume attacks are executed by a follower in order to study the potential impact of attacks. Another argument for this is that leader vehicles can be placed under additional scrutiny and replaced if necessary, while followers are more difficult to replace if they are non-cooperative. For example, if the platoon detects that the leader misbehaves, the vehicles can decide to leave the platoon and join a new one using the associated protocols for platoon management [121].

Jamming

Denial of service (DoS) attacks include a variety of jamming attacks (regular and reactive), which are the most popular variant of this attack within vehicular networks. This also includes various attacks on higher layers that influence reception (e.g., violating the MAC protocol, causing others to wait). Some authors [123] also suggest a DoS attack may be launched on the network layer, in order to disrupt communication by overloading the hardware security module (HSM). This leads to a form of *cryptographic packet loss*, i.e., packet loss caused because messages cannot be verified, or cannot be verified in time [91]. In this work, we consider all of these attacks to have the same effect, i.e., that messages will no longer arrive, and we refer to it as jamming. This generalizes the attack compared to the related work [150], where the authors considered reactive jamming.

V2X Data Injection

The second class of attacks on CACC that we identify is V2X Data Injection, which generalizes a variety of different attacks discussed in related work [123, 127]. Any attack that requires the attacker (or a vehicle compromised by the attacker) to send modified packets falls into this category; it is particularly interesting for this thesis because these types of attacks may be feasible to mitigate through misbehavior detection. The specific implementation of this attack is usually dependent on the type of data exchanged by the controller, and thus the corresponding impact is different for every controller. Similarly, the goal of an attacker is significant for the implementation, and sometimes used as a distinguishing factor in the literature (this includes the reduced headway attack, the collision induction attack and the misreporting attack, among others). However, these goals can often be achieved through one of the other classes of attacks (e.g., collision induction is possible through jamming), and thus we avoid this terminology.

Spoofing and replay attacks on the V2X messages can also be considered examples of this class, but these are considered mostly solved in the vehicular networking security community. In vehicular networks, where GPS is often assumed to be practically ubiquitous, replay attacks can be protected against using (close to) synchronized clocks. For this work, we assume that GPS spoofing is out of scope, since this type of attack is completely separate from the network, and such an attack would affect all vehicles in the vicinity in the same way. There are detection mechanisms available for this attack [147], and future global navigation satellite systems are designed to have cryptographically protected signals, greatly increasing the difficulty of spoofing attacks. Similarly, because cryptographic keys are authenticated through pseudonymous certificates, and messages are protected by digital signatures, spoofing attacks that cause the receiver to misinterpret the origin of the message are not possible. This also applies to message falsification attacks (i.e., attacks that manipulate messages of other vehicles), for which digital signatures are an adequate countermeasure. However, it is possible to use multiple certificates to simulate more than one vehicle – this is referred to as a Sybil attack (as originally introduced by Douceur [9]). Whether this attack is feasible depends on how easy it is for the attacker to obtain multiple pseudonymous certificates that can be used simultaneously.

Sensor Manipulation

The third and final class of attacks is sensor manipulation, which constitutes any attack that causes the internal network of the vehicle to report incorrect information to the controller. An attacker could have access to the internal network of the vehicle and transmit false data here, causing the vehicle to react incorrectly (or even destabilize the platoon through the controller attack [126]). Although this could also be done in software, i.e., manipulating memory before the controller processes the inputs, attacks on the internal network are considered to be more interesting, because misbehavior detection is a potential countermeasure. Software attacks, on the other hand, are difficult to prevent through misbehavior detection, as the detection system will likely also be compromised in such a setting.

Another variant of sensor manipulation positions the attacker outside of the target vehicle: instead of attacking the network, the attacker uses their sensors and other tools to blind the sensors of the target vehicle, or cause them to behave in a certain way. Researchers have shown that both LIDAR and cameras are vulnerable to such attacks [135], and the effect of such an attack is very similar to the other type of sensor manipulation attack (i.e., the controller sees invalid inputs). The only distinction is where the attacker is located, which influences the cost of such an attack, but the expected results for the platoon behavior are similar, and thus for the study of attacks we place these attacks in the same category.

7.4 Attack analysis on CACC

In this section, we study cooperative adaptive cruise control (CACC), one of the most promising C-ITS applications, aimed at increasing road efficiency through partially autonomous driving. CACC is essentially an extension of existing cruise control (CC) and adaptive cruise control (ACC) technologies, both of which are designed to allow the driver to maintain constant speed while driving. In the case of ACC, vehicular sensors, such as RADAR, LIDAR and cameras, are used to measure the distance to the preceding vehicle, in order to automatically respond to changes in its driving behavior. CACC extends this concept by allowing vehicles to communicate and create a *platoon*, consisting of a leader vehicle and multiple followers, as shown in Figure 7.3. There are many different controller implementations in the literature that enable this behavior [143] since the proposal of CACC in the 90s. It has been shown that CACC can be more efficient than ACC, in terms of energy usage and road utilization. It has also been shown that even a constant spacing between vehicles (i.e., independent of platoon speed) can be theoretically achieved, whereas it has been shown that ACC alone cannot achieve this goal [94].

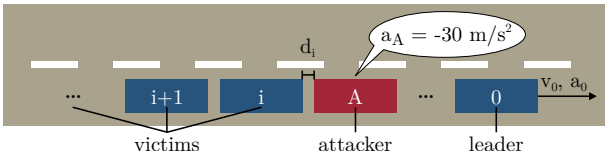


Figure 7.3: This figure shows a platoon with a leader and several followers: in our work, the red vehicle is the attacker, and the platoon length is 8. The attacker transmits a malicious beacon with a false acceleration of -30m/s^2.

Existing work shows that attacks can impact safety (i.e., potentially cause accidents), but lack a rigorous analysis and a framework within which such attacks and the resilience of the respective controllers can be analyzed. In this attack analysis, we lay the groundwork for such an evaluation framework: we implement several attacks and provide a concrete quantification to measure the impact of an attack. Going beyond simply finding whether an accident occurs or not, we analyze the significance of such an attack (in terms of the Δv of the accident) and quantify how the stability of an individual platoon is affected even if no accident occurs. We present results for three existing controllers implemented in *Plexe* [122], a state of the art simulation toolkit based on Veins, which simulates both controllers and network communication. Because we use this extensible platooning simulator as a basis for our evaluation, our work can easily be extended with new controllers. Similarly, our attack implementation can easily be extended to study the impact of more attacks. We publish the source code, data analysis scripts, and recorded data in order to stimulate further research in this area.

7.4.1 Related Work on Platooning Security

We distinguish two areas of related work: controllers, which implement the functionality that CACC and ACC offer, and existing studies on misbehavior in platooning and similar settings.

Controllers

Platoons are vehicles driving in a group, attempting to keep a minimal but still safe distance between the vehicles within the group. This can be achieved using ACC and CACC; the main reason for using CACC is that the safe distance can be much smaller. In both settings, a platoon normally consists of a single leader and a set of followers, together referred to as the platoon (or sometimes string). The set of followers can be dynamic, although this usually requires some kind of join and leave procedures. Each vehicle implements an upper controller that determines the desired acceleration, which is used to configure the vehicles' engine and brake controller. In this work we focus on the upper controller (i.e., setting the desired acceleration) exclusively, since this is the controller that an attacker can easily reach (either through communication or through attacks on sensors).

There is a broad scope of literature from the control theory and engineering side covering a variety of different controllers [143]. This literature typically introduces two stability metrics: local stability and string stability. Here, local stability refers to a vehicles' distance to the vehicle ahead of it, while string stability is a property of the platoon that determines how spacing errors propagate through the platoon. A platoon is considered string stable when the ∞-norm of the transfer function is less than or equal to 1 ($||\hat{H}(s)||_\infty \leq 1$) [94], which basically means that the platoon will eventually converge to a stable state if the leader keeps a specific velocity perfectly. Maintaining the distance to the next vehicle can done by keeping a fixed distance (i.e., meters) or by maintaining a speed-dependent headway time (i.e., seconds), referred to as constant spacing and constant-time spacing. Earlier work has shown that non-cooperative methods (i.e., ACC) cannot achieve string stability in a constant spacing policy [94].

In this work we focus on controllers implemented by Segata et al. [122] in Plexe, which includes a constant spacing controller, a controller with graceful degradation and a consensus controller. The constant spacing controller is taken from Rajamani [94] and keeps a fixed distance based on the measured distance and the received speed and acceleration of the preceding vehicle and the leader vehicle. The controller by Ploeg et al. [107] is designed to enable graceful degradation on network errors: their degraded CACC controller predicts the acceleration of the preceding vehicle if this information is not received. The idea is that this degraded controller outperforms ACC, and can thus be used to bridge communication gaps; this is particularly interesting for jamming attacks. The last controller is a consensus algorithm by di Bernardo et al. [124], which uses the position, speed, and acceleration information of all vehicles in the platoon. This controller has a larger spacing, but aims to maintain stability bidirectionally (i.e., vehicles also adapt their behavior to that of vehicles behind them); this is particularly interesting with respect to data injection attacks.

Existing Attacks

Attacks on CACC have typically been studied from two different perspectives, mostly working in parallel. Security research in vehicular networks has already suggested new types of attacks for the overall network in the previous decade [24]. More recent work has concentrated on misbehavior detection [141] and data injection attacks, which are uniquely suitable for vehicular networks and other cyber-physical systems. Platooning is one of the motivating examples for these researchers. On the other hand, authors from the control theory side have gained an understanding of potential security concerns, and have started developing attack-resilient control algorithms. In terms of attacks, some authors have also investigated how to exploit knowledge of the controller to show that attacks causing instability are possible.

Alipour-Fanid et al. [150] discussed platoon behavior under jamming. In their attack scenario, the jammer is a drone flying above the platoon, aiming to use its limited power to disrupt the platoon from above. Jamming is thus done reactively (i.e., with success rate $< 100\%$) to conserve power, although power consumption is not considered further. Their work shows that their ACC controller is string stable at 2.2 seconds headway distance, while CACC under the influence of their reactive jammer is string stable at 1.7 seconds headway distance. Additionally, they show the attack works best when the attacker is located near the first vehicle in the platoon.

Amoozadeh et al. [123] discuss a two-level controller model and describe a variety of attacks, including application and network layer attacks, as well as other issues such as sensor tampering and privacy issues. They present a result of a message falsification attack, which in their setting is a vehicle external to the platoon that falsifies all beacons; the result of the attack is platoon instability (i.e., the platoon does not converge back to a stable state). They also explain that such an attack is most effective when acceleration changes occur. Similar results are presented for a jamming attack – here, the authors' controller downgrades to ACC automatically. In our work, we aim to show that such a downgrade is not necessarily sufficient, depending on how system parameters are chosen.

DeBruhl et al. [127] also use a two component controller, consisting of a PD feedback controller (radar-based) to keep distance and a feedforward controller (communication-based), whose outputs are added together to form the desired acceleration that is provided to the vehicle drivetrain. This controller has been shown to be string-stable in real networked platoons, and DeBruhl et al. examine its behavior under various attack strategies, including a collision induction attack. The authors continue to develop an error calculation and detection algorithm, which essentially estimates the expected behavior of the vehicle in front and switches to ACC if this vehicle appears to behave differently than it claims to.

Outside of the context of communication, but also relevant to secure and safe platooning, is the work from the research group of Gerdes et al. [101] and Dadras et al. [126]. These authors looked at ACC-based platoons, and examined adversarial behavior within such a platoon, where the attacker systematically disrupts the distributed control algorithm using another algorithm that manipulates control input of their own vehicle to achieve their goal. In the first work, the authors show that for a given PID controller, they can cause significantly increased energy consumption in neighboring vehicles; as reduced energy consumption is one of the main drivers of CACC research, this impact is significant. The second work shows that for a PD controller, the attacker can cause the platoon to be asymptotically unstable, i.e., destabilize and eventually dissolve the platoon.

As mentioned, a rigorous and repeatable analysis of attacks across multiple controllers has not yet been performed. In our analysis, we provide the ground work for such an analysis, comparing different controllers and different attacks. As expected, we notice that different attacks affect different controllers, suggesting that there is indeed a need for more rigorous comparisons between controllers.

7.4.2 Simulation Setup

We use Plexe (version 2.0) [122], which is a simulation toolkit based on *VEINS*, as basis for our analysis due to its flexibility in implementing control algorithms and accurate simulation of network behavior. Plexe also extends SUMO, which provides realistic microsimulations of physical vehicles; it extends this toolkit with various controllers: a constant spacing controller (CACC) [94], a loss-tolerant controller (Ploeg) [107] and a consensus controller [124]. Note that Plexe does not implement the complete degraded CACC mode discussed in the original paper, due to limited sensing capabilities. For our implementation, we leave Plexe mostly unchanged, and only extend it to include our attacks. We implement four attacks: one jamming attack and three data injection attacks.

Our jamming attack is designed to analyze the maximum possible impact of an attack, and is therefore implemented by locally dropping received messages after the attack starts. This simulates any type of denial of service, including cryptographic packet loss and various jamming strategies, and provides an upper bound on the attack success, although it may overestimate it in some cases. The alternative would be to implement a specific attack (e.g., as done by Alipour-Fanid et al. [150]), but this approach is not as generalizable as ours.

Our data injection attacks are all implemented by exchanging some data with false data in the message an attacker would send according to the normal control algorithm. We analyze three different types: injecting false positions, false speeds or false acceleration values. There are multiple ways to implement these attacks; in our case, we set the attackers' value to a constant value for speed and acceleration, which greatly simplifies the interpretation of the results. In addition, we again aim to provide an upper bound on the attack effectiveness. For position falsification, we provide a position error that increases linearly over time, and is added to the vehicles' current position for each beacon.

The simulation scenario we analyze is a standard scenario from Plexe, which was used by the authors to analyze platoon behavior: the sinusoidal scenario. This scenario initializes the platoon and then provides the leader with a specific acceleration profile that leads to a sinusoidal speed graph. By performing attacks at different points in time, we can use this scenario to approximate the impact of

Table 7.4: Excerpt of simulation parameters: full parameters available in the data and source code repositories: `https://osf.io/fr7h5/`.

transmit power	100mW
sensitivity	-94dBm
platoon length	8
attacker ID	3
controllers	CACC, Ploeg, Consensus
CACC spacing	5, 7, 9, 11, 13, 20 m
target speed	50, 80, 100, 120, 150 km/h
attack: speed value	-50, 0, 50, 100, 150 m/s
attack: accel value	-30, -10, 0, 10, 30 m/s^2
attack: position shift	3, 5, 7, 9, 11 m/s

attacks on real behavior. The significant simulation parameters can be found in Table 7.4; the complete configuration file and code is available on Github[5]. Because of the way Plexe is implemented, simulating chain collisions reliably was not possible, so we collected information about the first collision only.

7.4.3 Results

This section presents the results of our attack experiments, and shows how we used our metrics to gain insight into the effectiveness of different attacks, as well as the resilience of the existing CACC algorithms. The data on which these results are based, are also published on Github[6], although they can also be re-generated using the source code and configuration mentioned above.

Jamming

We first discuss the results of our jamming attack, which are shown in Figure 7.4. This figure includes both successful attacks (i.e., crashes, marked in red) and stability impact (i.e., no crash, marked in blue), as well as the corresponding impact (see Section 7.2.4). The figure shows aggregated values over five runs. In all cases, the impact of the attack remained similar (i.e. all five runs resulted in collisions or all five runs did not result in collisions). We simulated this for different *target speeds* and different *jamming times*: the target speed is the intended average speed of the platoon (i.e., the average speed of the leader vehicle), while the jamming time is the time at which jamming starts. The jamming time directly corresponds to the range in which acceleration of the leader vehicle is positive; since the leader data is disseminated throughout the platoon, the attacker can use this acceleration profile to choose the time of attack. Based on the speed profile generated from Plexe's standard configuration, we determined that the positive acceleration lies between 30 and 32.5 seconds of simulation time (since the behavior is sinusoidal, this pattern repeats every 5 seconds).

As can be seen in the figure, the target speed has no impact on the effectiveness of the attack on CACC. However, the attacks are more successful with higher velocities when Ploeg or Consensus are used as the spacing of these algorithms depends on the platoon target speed. We can therefore conclude, that spacing has a strong influence on attack effectiveness. On the other hand, attack effectiveness is not directly influenced by the target velocity. In general, with more spacing, attack impact decreases. The figure shows that this is not always true (e.g. CACC-13 vs. CACC-20 at 32s simulation time). This is due to the fact that with CACC-13 and lower, the attack occurs between the attacker (car 3) and the subsequent car 4. With CACC-20, car 4 can avoid colliding with the attacker, however, car 5 does collide with car 4.

Position Injection

The next step in our analysis covers position falsification attacks, often cited as one of the most significant types of data injection attacks on vehicular networks. In our model, the false position is a shift of the attackers' real position that increases over time. It turns out that this attack is completely ineffective

[5] `https://github.com/vs-uulm/vnc2017-CACC-code`
[6] `https://github.com/vs-uulm/vnc2017-CACC-data`

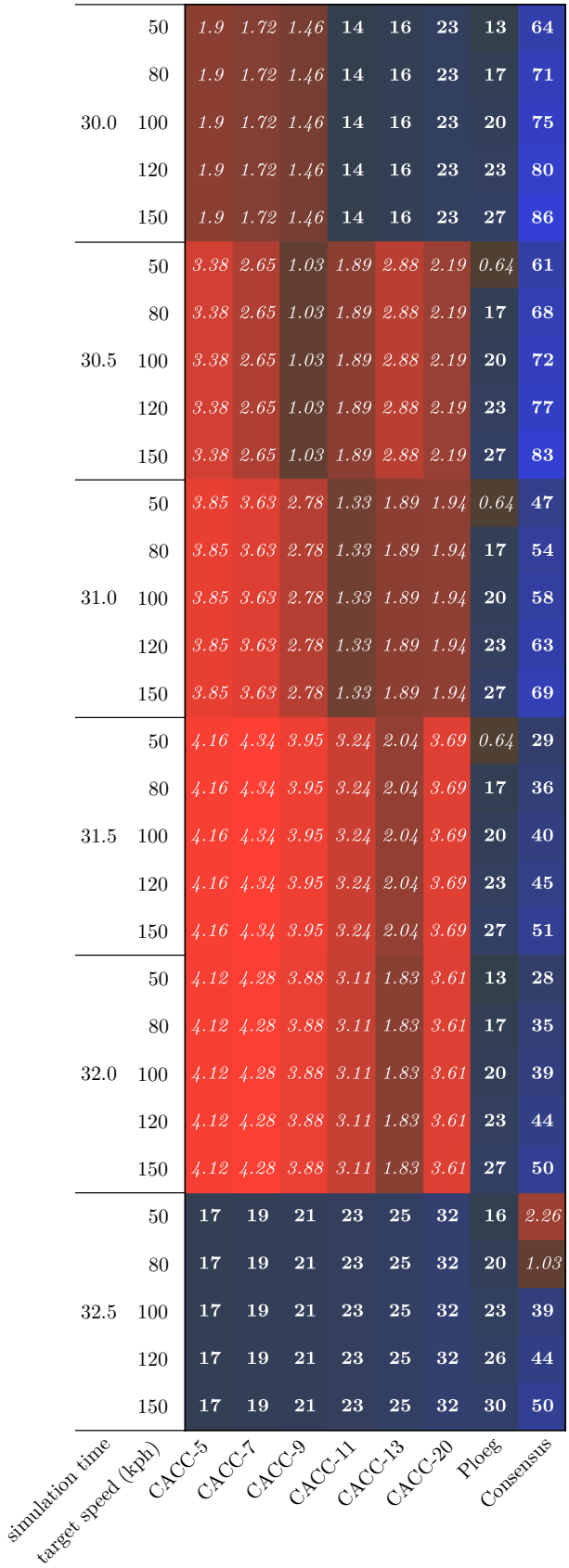


Figure 7.4: Heat map showing attack success and impact for the constant jamming attack at the given point in simulation time, for different target platoon speeds and different controllers. Red/italics: attack leads to accident; number is impact velocity in m/s. Blue/bold: no accident; number is maximum spacing error in m.

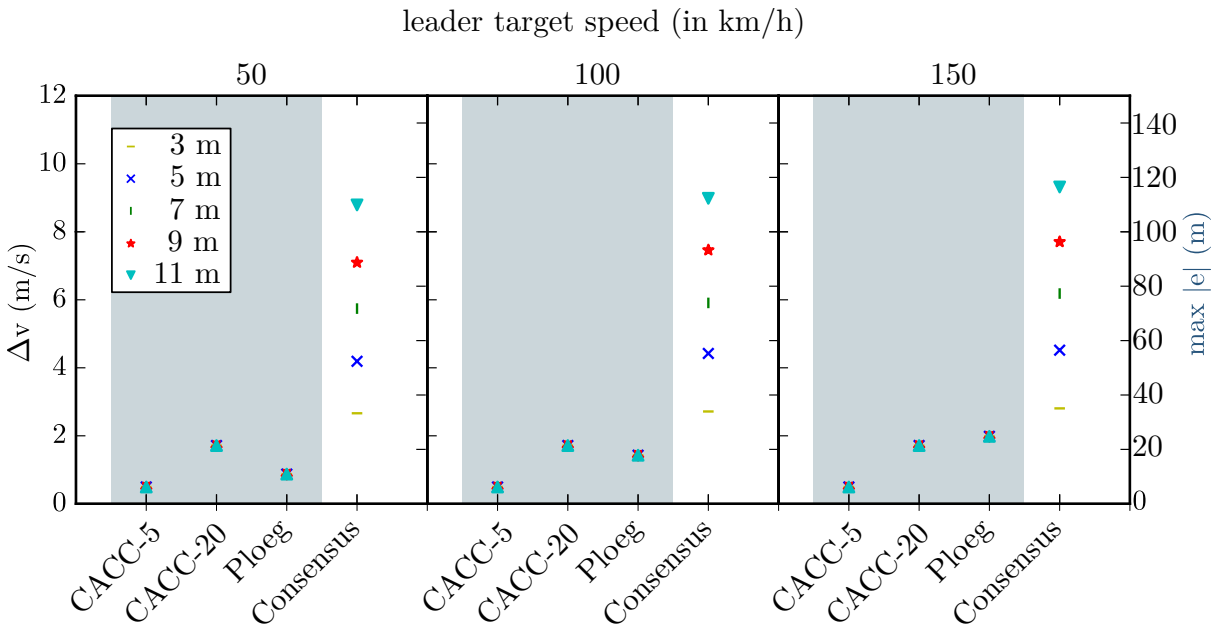


Figure 7.5: The effect of position falsification. The different symbols represent different attack magnitudes. The horizontal axis shows different controllers (bottom) and different platoon target speeds (top). The points with blue background were not associated with crashes; for these points, the maximum observed instability is plotted. For crashes, the Δv of first two crashing vehicles is shown.

against the Ploeg and constant spacing controllers – it only causes crashes when the consensus controller is used. However, we can furthermore see that the spacing error of the Ploeg controller increases significantly with the leader target speed. This is because the consensus controller is the only one that uses received position information directly. Somewhat surprisingly, the impact of this attack on the consensus controller depends only on the value transmitted by the attacker, and not significantly on the target speed with which the leader (and thus the platoon) is moving. The full results are shown in Figure 7.5: for each shift parameter (mark and color of the scatter plot), with different leader target speeds (top, left to right), the graph includes the results for four different controller configurations (bottom, left to right). The impact speed of successful attacks (white background) varies from 2 to 9 m/s, while the maximum spacing error (gray background) is unaffected by the position falsification. This means that all attacks that have any effect lead to an accident.

Speed Injection

Our next attack was manipulating the speed transmitted by the attacker. Here, we continuously output the same value regardless of platoon or controller behavior, which causes the error between the attacker's value and actual value to vary from 10 to -10 km/h if it falsifies the platoon target speed. The result of our analysis, displayed in Figure 7.6, shows that speed injection is successful against constant spacing CACC only, where the most significant crashes happen at lower speeds; the attacker claims to accelerate, but actually does not. When the attacker claims negative speeds (e.g., -50) the result is a high position error. The Ploeg controller shows the same behavior as before (i.e., it is not affected by the attack), and here the consensus controller is also unaffected, although the spacing error is much higher. Due to a constant falsified speed, the attacker does not achieve success in the constant value matching the leaders' speed; we expect this will change if the leader behavior is less predictable.

Acceleration Injection

The final attack we performed is manipulation of the transmitted acceleration. Here, we also chose a relatively extreme scope, in order to discover the maximum potential of these attacks ($-30, -10, 0, 10, 30$ m/s^2). The resulting statistics are shown in Figure 7.7. As expected, this attack affects all controllers, because acceleration is used in all the control algorithms. However, the effect is widely different per

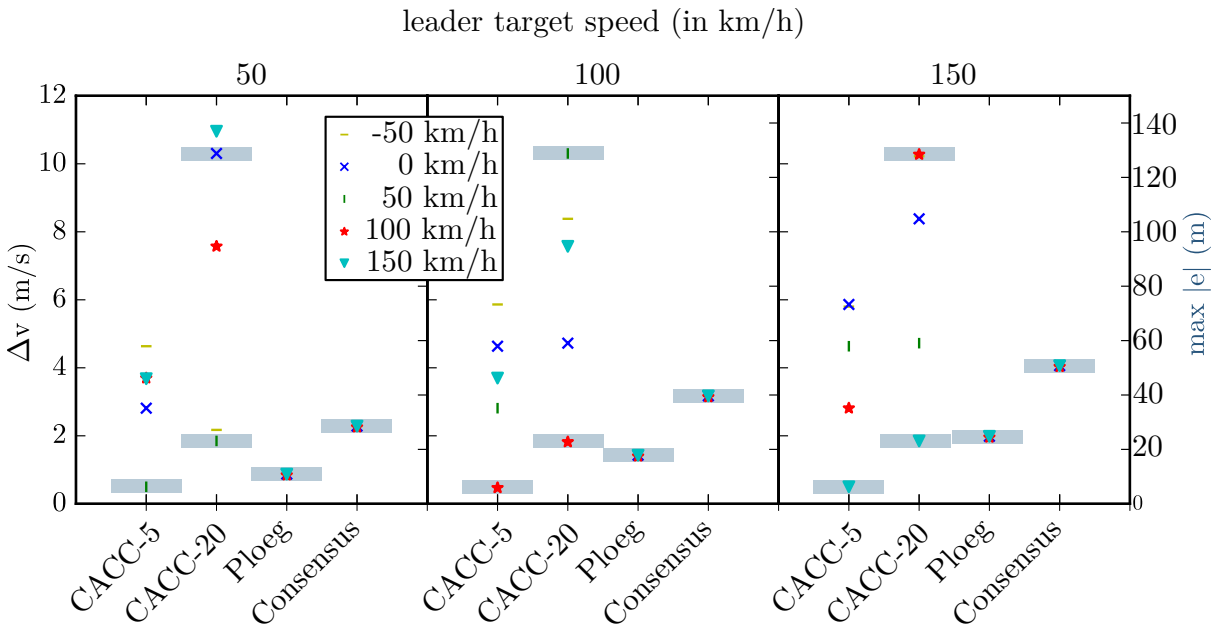


Figure 7.6: Attacking with false speeds. The different symbols represent different attack magnitudes. The horizontal axis shows different controllers (bottom) and different platoon target speeds (top). The points with blue background were not associated with crashes; for these points, the maximum observed instability is plotted. For crashes, the Δv of first two crashing vehicles is shown.

controller: the consensus controller can tolerate these large errors in acceleration information (although the maximum spacing error increases slightly), while the others fail almost completely. This is likely due to the fact that the consensus controller is the only bi-directionally coupled controller (or more precisely, each vehicle considers all others in its control algorithm). Similar to the speed injection attack, injecting false acceleration data causes attack impact dependent on the injected value; the attack impact is significant for positive acceleration values. However, here the impact of false negative acceleration is also significant — however, at some point, the distance sensor notices that the distance is too large, and it resets to ACC behavior.

7.5 Discussion

Our results show that the CACC and consensus controllers are unreliable when subject to effective denial of service attacks, and that the impact of injection is also significant for any data that is used in the controller directly. This confirms earlier results on individual jamming and injection experiments, and shows that these results also hold when simulation parameters are changed. Beyond this, our results show that the window of opportunity for the attacker to jam or transmit false information is broad, i.e., the attacker does not need to hit precisely the correct timing for the attack to work. Our results can also be used as a first step for an attacker to increase attack effectiveness, which enables more a rigorous analysis of more resilient control algorithms.

Overall, we can conclude that the Ploeg controller deals well with jamming even considering that this controller was only intended for intermittent, unreliable channels, and not for complete failure. However, we still managed to cause some accidents even with this algorithm, suggesting that an eventual complete fallback to ACC may be advisable. The consensus controller behaves differently, and only fails with a later starting point when jamming — future work could more thoroughly investigate the cause of this behavior. Finally, we conclude that the constant spacing controller is completely unsuitable for unreliable communication; the window of opportunity within which a crash can be triggered is at least 2 seconds.

For data injection attacks, the actual value transmitted by the attacker had an important influence over the attack impact, but the target speed of the leader did not matter much in comparison. We expected our attacks to increase in impact as the overall platoon speed increases, because we expected

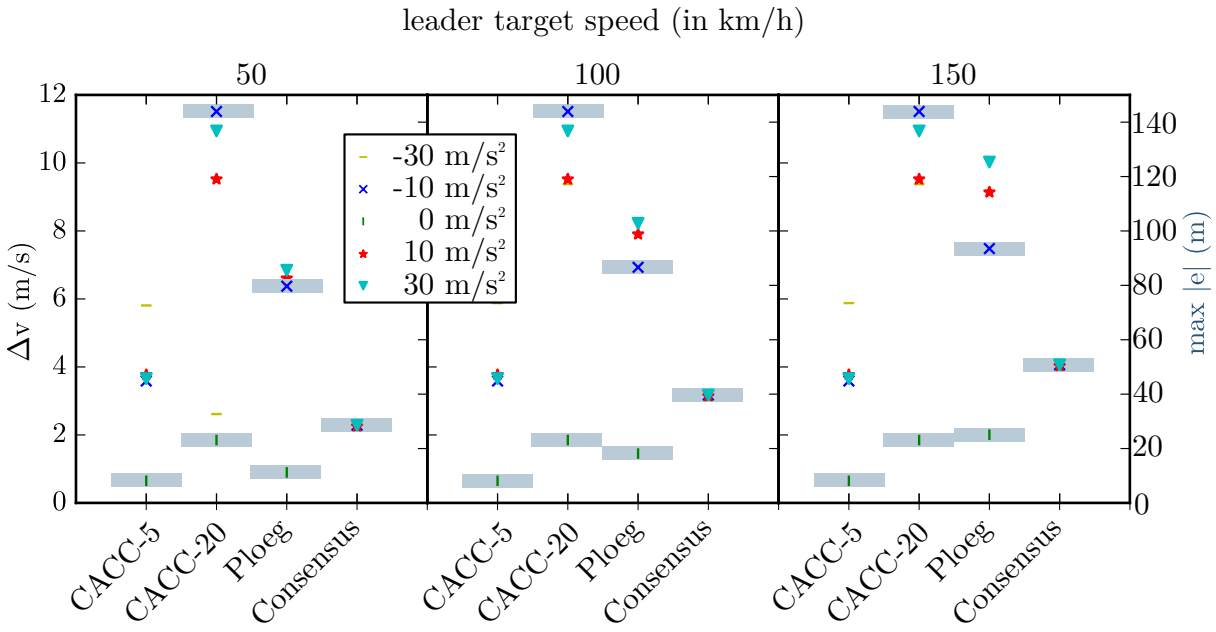


Figure 7.7: Behavior of controllers with falsified acceleration information. The different symbols represent different attack magnitudes. The horizontal axis shows different controllers (bottom) and different platoon target speeds (top). The points with blue background were not associated with crashes; for these points, the maximum observed instability is plotted. For crashes, the Δv of first two crashing vehicles is shown.

the controllers to react more quickly in those scenarios. Although this is the case for some position errors, the crash impact remains relatively consistent for the CACC controller. For the Ploeg controller, the impact is dependent on the speed, but this dependence is mainly due to the speed-dependent gap that should occur between vehicles. The one exception to all these observations is the consensus controller, which is surprisingly resilient against attacks, showing no impact from speed and acceleration falsification. However, the impact of position shift on this controller is significant. Nevertheless, this suggests that the consensus controller might be a good starting point for building a resilient controller, despite its relative ineffectiveness, illustrated by a higher overall position error.

Our results are likely to extend to similar scenarios; the CACC and Ploeg controllers both rely only on the leader and the preceding vehicle, so any platoon long enough to contain the crashing vehicle is affected by potential crashes. It is unclear how well the results transfer for the consensus controller, since its intended behavior is quite different from the other two controllers, considering all the vehicles in the platoon for the control of each vehicle. However, the overall methodology we applied here is not constrained to the scenario — the tools we provide can easily be used to further tune the parameters of the controllers for increased safety, or the attackers' parameters to improve attack effectiveness.

7.5.1 Concluding the Attack Analysis

In this attack analysis, we have discussed a more holistic attacker model, implemented several attacks and tested their impact on existing controllers using our proposed analysis methodology. Overall, the results show not only that controllers are vulnerable to attack, but that there are significant differences between controllers in terms of how they are affected by attacks. The research community should critically discuss the limitations of CACC under the influence of these attacks. Work should therefore focus also on prevention, through the design of resilient controllers on the one hand, and effective detection and revocation mechanisms on the other.

To further improve false data injection attacks, one could look at how other types of controllers behave, or study the effectiveness of simultaneously falsifying multiple fields in the beacon. The latter is particularly relevant, because it circumvents many typical misbehavior detection mechanisms (which could detect most of the attacks we have described with reasonable accuracy, in part due to the values

we chose). An alternative route that has not yet been studied very well is the injection of false data into external sensors. Some research has shown that camera and LIDAR sensors can be tricked easily in a lab environment [135]. Once Plexe has been extended to include some sensor error models, experimenting with these attacks and combining them with false data injection will likely be an interesting path. Even though attacks on sensors would rely on line-of-sight and additional equipment (i.e., it cannot be done by manipulating a vehicle with malware), we have also found that some injection attacks cause crashes between vehicles further behind the attacker.

For the development of future control algorithms, we suggest that authors consider a fallback mechanism similar to the Ploeg controller. However, we suggest that this eventually downgrades from degraded CACC to pure ACC, because the error in the estimated acceleration will increase over time if a jamming attack is maintained long enough. To improve reliability, using all available inputs may be a feasible approach to reduce the impact of attacks (e.g., as partially demonstrated by the consensus controller), especially because common controllers typically only use leader and preceding vehicle information. This may not increase overall security, but will make it more difficult to successfully fine-tune attacks and achieve a desired impact. Should the information show inconsistencies, a fallback mechanism could again be considered; here the challenge is ensuring that the entire platoon mutually decides on performing a fallback, to avoid a heterogeneous controller environment.

The attacks discussed in this section may be detected with relative ease using plausibility checks; however, the main aim of this analysis was to show how attack impact may be maximized. Future work in this specific area may look at the maximum tolerance of the attacks we designed; however, our currently proposed metrics are bound to the behavior of the leader. The next step is then to decide how to use this information, apart from invalidating the vehicle in the SCMS [168], preventing damage from ongoing attacks is also a goal. If using a controller with graceful degradation (similar to Ploeg), the detection may trigger this functionality: when combined with effective revocation, this will ensure that temporarily degraded service is the best possible attack. An interesting side aspect is that not all detection algorithms are compatible with platooning, because the corresponding behavior is different: it may thus be a feasible path to design specialized misbehavior detection mechanisms for the CACC scenario specifically.

For the remainder of this thesis, we aim at an analysis of detection performance that is not dependent on a specific attack. Although it may be easy to design detection mechanisms for these specific attacks, the likely more feasible approach is to build a controller that is more resilient to faults. This is because a detector that models the behavior (i.e., positions over time) of a platoon is necessarily an approximation of the controller's intended behavior. Although cross-validation between received V2X data and objects observed through sensors is in principle a way to detect potential attacks, we do not have the tools to evaluate such a system. This is because neither Plexe, nor any other open-source simulation toolkit that we are aware of, is capable of simulating realistic sensing behavior. Any reasonable detection mechanism that we are aware of that could help in these types of scenarios, particularly against realistic attacks (e.g., adaptations to existing work [75]) requires some kind of sensing information. Therefore, we focus instead on the second part of the evaluation foreseen by our methodology, which is the effect of deploying misbehavior detection in widely varying scenarios. This is important, since misbehavior detection should be a type of system that is always active, and evaluating small scenarios increases the risk of bias in the experimental design. This may in turn imply that the results do not generalize, as we found was the case with the overlapping position detector (see Section 6.1.4).

Chapter 8

Evaluation of Data-centric Detectors

In this chapter, we discuss the evaluation of individual data-centric detectors, which are used in the next chapter to assess fusion performance. One major contribution that also appears in this chapter is the VeReMi dataset, which is a public dataset designed for the evaluation of data-centric detectors. As far as we are aware, this dataset is the first of its kind (in fact, the first publicly available evaluation dataset for misbehavior detection in this field in general). We also make contributions to rigorous evaluation of detectors, as introduced in Chapter 7. Parts of this chapter have previously been published [172].

8.1 VeReMi Evaluation Dataset

The first contribution discussed in this chapter is a dataset intended to provide a common baseline for misbehavior detection evaluation. Previous studies have always relied on individually designed simulation studies: although this has the advantage of customizable attacks and flexibility with respect to the specifics of the scenario, it makes it difficult to compare mechanisms with each other. The purpose of our dataset is to provide an initial baseline with which detection mechanisms can be compared. This reduces the time required for researchers to perform high-quality simulation studies, and it makes it easier for readers to compare the results of different papers. We acknowledge that no dataset can completely replace a detailed analysis of a detection mechanism; however, the current state of the art, where a comparison with any other scheme requires a time-intensive and error-prone re-implementation of every scheme, is unacceptable. Our dataset will provide the first step towards a comprehensive evaluation methodology for this field.

The dataset has advantages beyond being publicly available: it can easily be reproduced using the publicly available source code, and is therefore also extensible. This means that new attacks can be introduced into the dataset in a consistent way, without error-prone manual editing of the dataset. However, this advantage is very closely tied to the fact that VeReMi is based on simulative evaluations. We discuss the advantages and disadvantages of this approach in this section.

The dataset we introduce is based on our evaluation workflow, as previously discussed in Section 4.4.2; recall that this workflow in essence performs off-line classification for every vehicle, and then assesses the overall performance of a set of detectors. This process is illustrated again in Figure 8.1: in this chapter, we describe VeReMi and the associated components, highlighted in bold in this figure.

VeReMi essentially consists of message logs for every vehicle in the simulation and a ground truth file that specifies the attacker's behavior. Local information from the vehicle is included through messages of a different type (representing periodic messages from a GPS module in the vehicle). Any detector can thus read the sequence of messages and determine the reliability of every message (or a subset thereof). Our dataset and the source code used to generate it[1] is publicly available, and consists of 225 individual simulations, with 5 different attackers, 3 different attacker densities, 3 different traffic densities, and 5 repetitions for each parameter set (with different random seeds). A detailed discussion of these aspects and the choices made in the generation process is provided below. Note that anyone can reproduce and extend our dataset in a consistent way using the provided source code, enabling anyone to extend the evaluation of any detector that was studied with VeReMi.

[1] `https://veremi-dataset.github.io/`

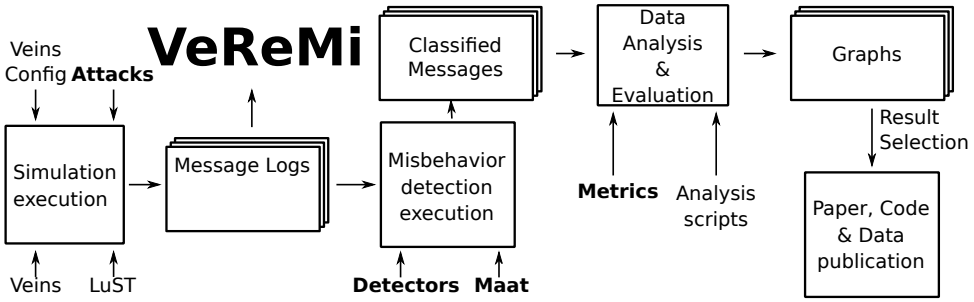


Figure 8.1: VeReMi's workflow – the detectors on the left side of the graphic can simply be replaced by new detectors; the simulation can be replaced by new simulations, resulting in an updated version of VeReMi.

8.1.1 Scenario Selection

The purpose of our dataset is to provide a holistic basis for evaluation of misbehavior detection mechanisms, rather than a specific traffic situation that works well or poorly for a specific mechanism. This is aimed to reduce unintentional selection bias based on properties of the mechanism and the scenario, sacrificing the level of detail with which individual scenarios are studied. The alternative approach that is often taken is to pick a few specific traffic scenarios to be studied (e.g., congested highways, free-flowing traffic in a Manhattan grid setting) and analyze these in detail. This provides detailed information on mechanism behavior, but relies on a lot of manual decision making, making fair comparisons between mechanisms difficult. We instead focus on how mechanisms behave in a variety of different scenarios. To this end, we provide a much larger dataset that can be used to assess the overall performance, before looking at individual scenarios to provide specific improvements for specific detection mechanisms. In order to achieve this, we selected a representative sample of the entire simulation scenario, based on the included road types and the associated traffic densities.

Our work is based on the Luxembourg traffic scenario (LuST), originally introduced by Codeca et al. [125], who aimed to provide a comprehensive scenario for evaluation of VANET applications. Although this scenario is very suitable for traffic engineering, the simulation cost for the simulation of a city-scale VANET over multiple hours is prohibitive[2]. For this reason, reproduction of an entire study performed by other research groups is quite rare in our community – most papers that reference results from other articles are follow-up work. This is where our dataset comes in: it provides a simple message stream per vehicle, making it much easier to reproduce detection studies. Table 8.1 describes some core parameters of the simulation – more information can be obtained in the OMNeT++ configuration file in our source code.

We implemented an initial set of attacks associated with position falsification, the type of attack that is most well-studied in our field (and for which many mechanisms have been designed [170]). Rather than implement a broad set of attacks, we focused on this specific attack category to show the efficacy of our approach. Ultimately our goal is to evaluate Maat, and its primary goals include the detection of different types of attacks through fusion. However, fusion by definition cannot improve performance if the detectors and attacks are unrelated. For example, if a detector can only detect false positions, an attack with false vehicle weight would never be detected[3]. We foresee that other researchers can contribute new attack implementations and corresponding datasets to the central VeReMi repository, which we will maintain. By focusing on a specific set of attacks in this work, we show how VeReMi is useful for other researchers and provide an initial starting point for the community. Since the data is published as a list of message logs, which include speed, claimed transmission time, reception time, position, and received signal strength indication (RSSI) for each receiver, it is easy to take a newer version of VeReMi and run it through detectors that have already been published. This enables researchers to directly compare their detector against existing ones, and any new attack against a variety of detectors (as long as their

[2]For illustration purposes; our 100 second excerpt of the scenario at high densities contains hundreds of vehicles and runs for a few hours – a significant part of this cost is the realistic simulation of signals reflecting off the ground and various buildings.

[3]To be precise, some detections may occur due to the detector issuing false positives. However, in this situation, attackers and benign vehicles will be statistically indistinguishable from each other.

Table 8.1: Simulation Parameters

Parameter	Value	Notes
Mobility	SUMO LuST (DUA static)	[125]
Simulation start	(3,5,7)h	controls density
Simulation duration	100s	
Attacker probability	(0.1, 0.2, 0.3)	attacker with this probability
Simulation Area	2300,5400-6300,6300	various road types
Signal interference model	Two-Ray Interference	VEINS default
Obstacle Shadowing	Simple	VEINS default
Fading	Jakes	VEINS default
Shadowing	Log-Normal	VEINS default
MAC implementation	802.11p	VEINS default
Thermal Noise	-110dbm	VEINS default
Transmit Power	20mW	VEINS default
Bit rate	6Mbps	VEINS default (best reception)
Sensitivity	-89dBm	VEINS default
Antenna Model	Monopole on roof	VEINS default
Beaconing Rate	1Hz	VEINS default

source code is published). Our attacks were previously discussed in Section 7.3.1.

8.1.2 Characteristics

The dataset consists of a total of 225 simulation executions, split into three density categories. The low density (corresponding to a run starting at 3:00) has 35 to 39 vehicles, while the medium density (a run at 5:00) has between 97 and 108 vehicles, and the high density (7:00) has between 491 and 519 vehicles. Out of these vehicles, a subset is malicious: this decision is made by sampling a uniform distribution ($[0, 1]$) and comparing it to the attacker fraction parameter, essentially assigning each vehicle to be an attacker with that probability. All of the vehicles classified as attacker execute the same attack algorithm (described in the previous section). Each receiver generates a reception log containing all periodic position updates generated by SUMO ($10Hz$) and all received messages (i.e., beacons from other vehicles). Each of these log entries contains a reception time stamp, the claimed transmission time, the claimed sender, a simulation-wide unique message ID, a position vector, a speed vector, the RSSI, a position noise vector and a speed noise vector. In addition, a ground truth file is updated whenever a message is sent by any vehicle: this file contains the transmission time, sender, attacker type, message ID, and actual position/speed vectors. The attacker type is set to 0 for legitimate vehicles. The following describes the dimensions of the VeReMi dataset in terms of messages and reception events per density.

The number of messages transmitted in the simulations varies between the simulations and densities; at low densities, 908 to 1144 messages are sent, at medium densities, there are between 3996 and 4489, and at high densities, there are 20482 to 21878 messages sent. The corresponding reception events are much more scattered; some vehicles receive 0 messages in these simulations (e.g., if they are not close to any other vehicles). For low density, a vehicle receives up to 278 reception events (total over all low density simulations is 277916 events spread over 2820 receivers), while at medium density this number goes up to 911 reception events (total over all 1815215 spread over 7695 receivers). Finally, for a high density, a single vehicle processes up to 5468 reception events in the 100 simulation seconds (total over all simulations over all 37500 vehicles is 37043160), or about 1000 messages per vehicle (10 per second, i.e., roughly 100 nearby vehicles at a beaconing rate of $1Hz$ if we ignore lost messages). A graphical view of reception event frequency is given in the histogram in Figure 8.2.

The scenario also includes a wide variety of traffic behavior, as illustrated in Figure 8.3, which shows aggregate speed statistics over all runs in a specific density. The statistics were computed by taking the current local speed vector for every vehicle for every position update (which happens at $10Hz$) and aggregating all these samples. This results in a mean speed of 24.36 m/s for the low density scenario, with a very large standard deviation of 13.73 m/s; since the median speed is 30 m/s, this suggests that most of the deviation is due to traffic lights. In the medium density configuration, the median (13.33

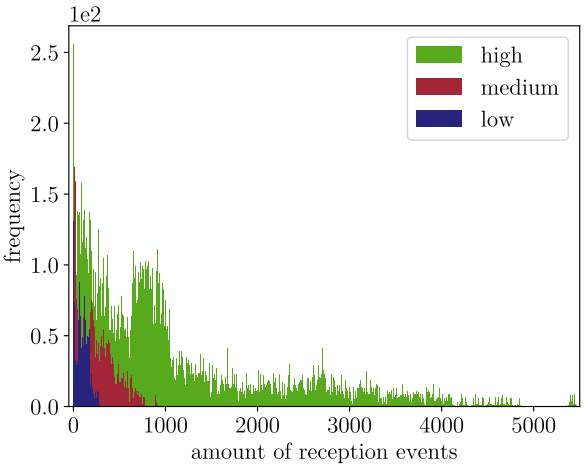


Figure 8.2: Histogram showing the raw number of reception events in the simulations.

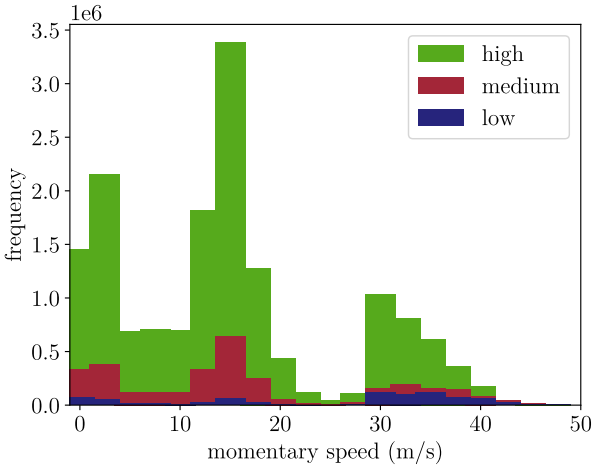


Figure 8.3: Histogram showing distribution of speed in the simulations.

m/s) and mean (15.06 m/s) drop significantly, although the number of vehicles in the simulation is fairly low (only about 2.5 times the vehicles compared to a low density); the standard deviation is still very high (12.34 m/s), indicative of the wide variety of driving behavior. Finally, our high density scenario drops down further to a mean speed of 12.81 m/s, with a standard deviation of 10.94 m/s, while the median is 12.81 m/s.

8.1.3 Limitations of VeReMi

Our dataset cannot be fully representative of all possible attacks in VANETs, especially because the implemented attacks are representative of a specific type of attack. Investigating the effect of multiple attack types across a single simulation is not possible with this dataset. We argue that our dataset should be used as a starting point for a more rigorous approach to the evaluation of such systems, primarily by motivating other researchers to also publish their data. In addition, other researchers can use this process to find weaknesses in our detection approaches and implement new attacks. Because these extensions would include the same base scenarios, but with different attacks, a user of the dataset should specify which attacks were used. The new attack can then be tested in existing detectors, either to choose new parameters for those detectors or to inspire the design of new detectors. We believe this interactive process is essential to achieving scientifically meaningful results: existing work nearly always relies on non-published code in some way, and thus it is very difficult to verify results of others. This leads to difficulty in replication of results, especially for complex detection systems that have many moving parts. The purpose of this dataset publication is to alleviate this: authors can make verifiable and reliable comparisons between their schemes and ours.

Another important limitation of our dataset is that the evaluation workflow, which was previously discussed in 4.4.2 is fundamentally non-interactive: it is designed for *detection*, not for *response*. This means that some specific misbehavior detection schemes that rely on interactivity or application decisions based on the detection of an attack (e.g., increasing safety distance in autonomous driving) cannot be evaluated with our dataset directly. However, for systems that protect specific applications, a comparison with other schemes always requires custom implementation. The core weakness of our approach is that we cannot directly evaluate trust over time without major modifications to our workflow, since trust schemes often do not output decisions for every message.

8.2 Evaluation of Plausibility Detectors

This section shows an application of our dataset and metrics to analyze several data-centric plausibility detectors, which are detectors that verify a received message against data from local sources only. The decisions made by these detectors are practically instant (i.e., they do not depend on other data sources),

Table 8.2: Detector parameters, chosen based on detector design and previous work in Chapter 6.

Detector	Parameter	Values
ART	est. reception range (m)	$100, 200, 300, 400, 450, 500, 550, 600, 700, 800$
SAW	max. appearance distance (m)	$25, 100, 200$
SSC	max. speed deviation (m/s)	$2.5, 5, 7.5, 10, 15, 20, 25$
DMV	min. distance (m)	$1, 5, 10, 15, 20, 25$

and contrary to trust schemes and consistency mechanisms, does not result in new attack vectors that can be used for *bad mouthing* and similar attacks [170]. As plausibility mechanisms are often used as a basis for trust establishment [30, 33, 35, 82], we focus on these. We implement detectors in our detection framework Maat[4]. In this chapter, we compare four detectors: the *acceptance range threshold* (ART), the *sudden appearance warning* (SAW), the *simple speed check* (SSC), and the *distance moved verifier* (DMV). Of these, the ART is the most well-studied, originally introduced by Leinmüller et al. [30] and later used by others, including Stübing et al. [82] and in our earlier work [141]. It basically uses the expected reception range as a measure for the plausibility of the position included in incoming single-hop beacon messages, which are the most important source of information for VANET applications. The SAW was also introduced by Schmidt et al. [35], and is based on the assumption that vehicles will not suddenly appear, but rather always approach from a distance; if a message originates close by with an unknown sender, it is considered malicious. The SSC and DMV were implemented as part of our work on a detection framework, and both examine whether a new beacon confirms information claimed in an older beacon. The SSC decides maliciousness based on how the claimed speed relates to the speed implied by the position and time differences between the current and the previous beacon, and the claimed speed in the current beacon. If the deviation exceeds a threshold, this detector classifies the message as malicious (similar to, but much simpler than, a Kalman filter [82]). Finally, the DMV checks whether the vehicle moved a minimum distance (similar to the way the MDM proposed by Schmidt et al. [35]), and if this distance is too small, the message is considered malicious.

This selection of mechanisms is made for several reasons: 1) all of these mechanisms are exceedingly simple, 2) these mechanisms are designed to detect false positions in some sense, but as our analysis will show, different mechanisms detect different attacks, 3) the mechanisms rely on different data elements in the packet. Especially the simplicity is important for this discussion, since this allows us to not only compare the mechanism performance dependent on their respective thresholds, but also showcase how our metrics and dataset enable a useful and detailed analysis of mechanism behavior. We also focus on position verification as a specific application, in order to focus on a specific set of attacks, as discussed previously. Finally, these are the mechanisms for which the source code is available, unlike other mechanisms we have found in the literature – re-implementing mechanisms can be challenging, as often the implementation details are missing due to space limitations, and the code is not publicly available.

8.3 Results: Detection Performance

Here we show the analysis results of our misbehavior detection framework, Maat, executing the detectors described above with different parameters, as listed in Table 8.2. In this chapter, we focus on the outcomes of individual detection mechanisms: for a real-world deployment of Maat, this evaluation process is the first step, and will inform the work in the next chapter, where we study fusion. These results can be used to configure initial thresholds for each detector. For brevity, we focus on high and low density scenarios with high numbers of attackers (30%), but the results are consistent with those for the other attacker fractions. We publish the entire set of figures and the underlying data as additional material[5]. The 10% and 20% attacker cases show comparable results for each set of graphs in Figures 8.4 and 8.5; similarly, the medium density is comparable to the high density (as the application behavior is quite similar, as illustrated by Figure 8.3).

As our dataset contains five simulation runs per behavior/attacker parameter set, each point in the graphs represents the mean of five runs, aggregated over vehicles as described previously. The error bars

[4] `https://github.com/vs-uulm/Maat`

[5] `https://github.com/vs-uulm/securecomm2018-misbehavior-evaluation`

in these graphs show the sample standard deviation associated with this mean. The colors show the different detectors, also listed in the legend on the bottom; for black-and-white readers, we point out that the extremes of the threshold values (indicated with arrows at the extreme ends of each plot) are unique. Finally, note that the lines in these graphs are for illustrative purposes only – as previously discussed, interpolation between these points is a non-linear task [16].

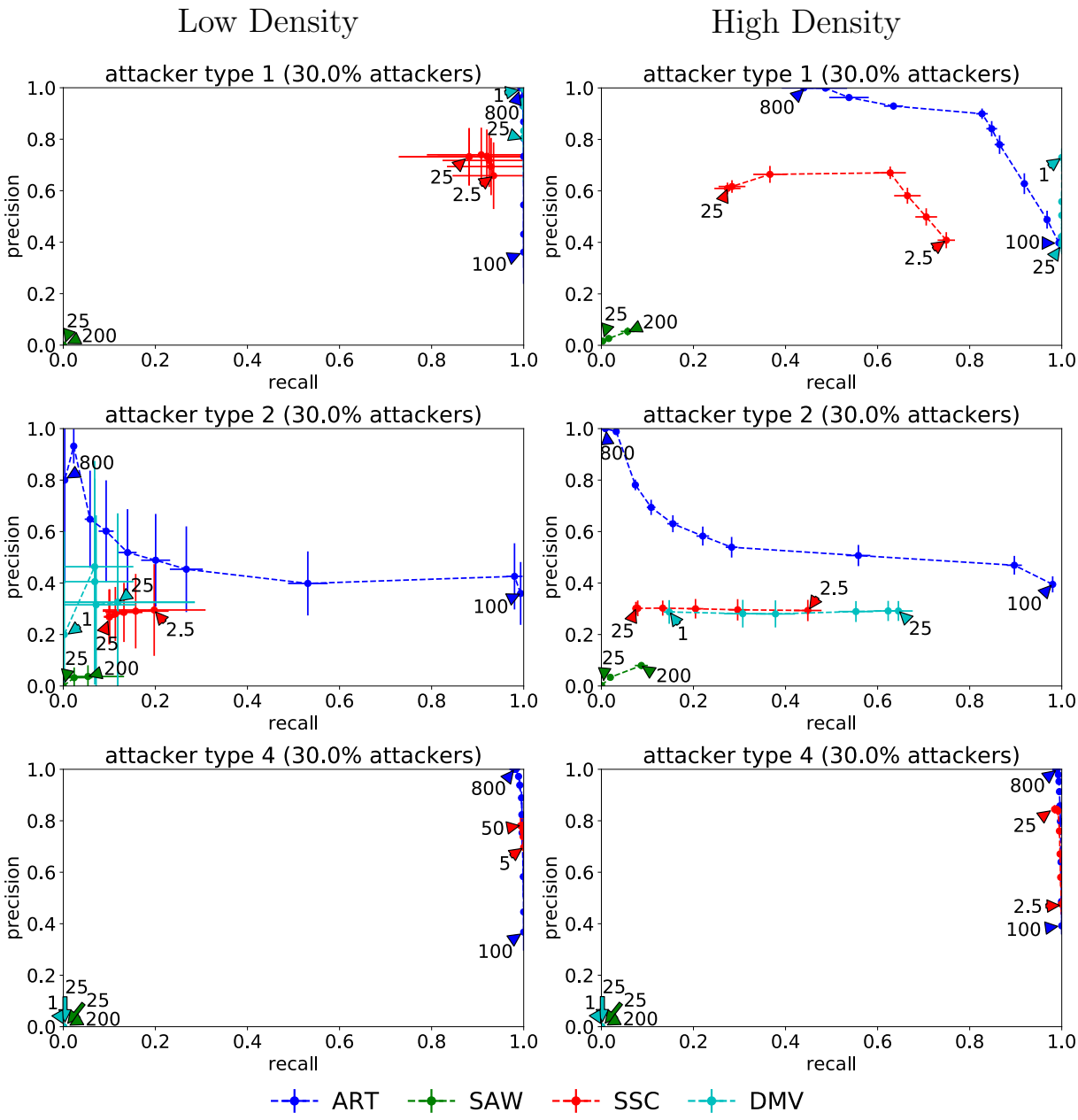


Figure 8.4: Precision-Recall graphs for low densities (left) and high densities (right) for attackers 1 (constant false position), 2 (constant offset of real position), and 4 (uniform random in simulation area).

8.4 Discussion: Detection Performance

In Figures 8.4 and 8.5, the different attackers are listed from top to bottom as specified in Table 7.3. Overall, as one might expect, the type of attack is an important distinguishing factor in the effectiveness of the detection process (i.e., easily detected attacks generally have higher recall). One can observe

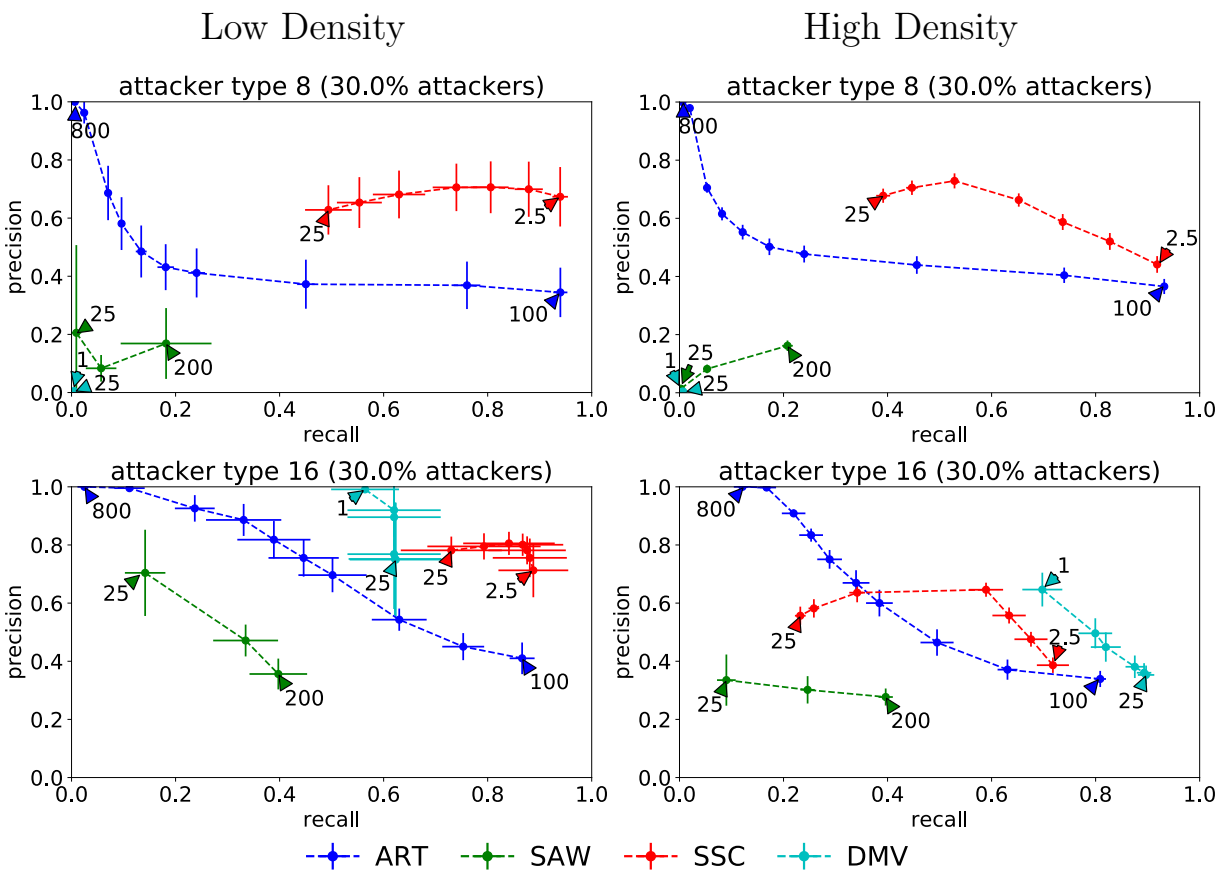


Figure 8.5: Precision-Recall graphs for low densities (left) and high densities (right) for attackers 8 (random offset for every message) and 16 (simulating eventual stop).

immediately that the results for the different detectors vary greatly per attack, regardless of density. However, some detectors' performance is dependent on the density of the traffic (DMV is notable here). It can also be seen that the SAW has very poor performance for all attacks except for 16 – this corresponds to expectations from the detector design. We now focus on a brief discussion of each individual attacker.

For the first attacker, which falsifies position, the results show that in low density settings, all detectors except SSC very accurately detect the attack. A similar trend can be observed in the high density scenario; however, note that the ART performs slightly worse at very high thresholds (greater than 500); this conforms with results from our earlier study [141], as discussed in Chapter 6, which used a different simulation model. With regards to the SSC, which verifies whether the claimed speed in the current beacon corresponds to the distance moved between two beacons. However, note that this is not necessarily correlated with an attack: it occurs naturally in the application behavior that vehicles' speed deviates significantly from the movement, for example when braking for traffic lights. Since no interpolation is performed by the SSC based on other information (such as sensor measurements) and the beacon frequency is relatively low, this mechanisms' performance is overall quite poor.

The second attacker, type 2, adds a fixed vector to its position; this attack is harder to detect for most mechanisms, and this can be observed by the poor performance in all cases. The very large standard deviation in the low density case (left) suggests that the success is very dependent on the relative position of the vehicles; especially for ART, this is exactly what one would expect. This is confirmed by the greatly reduced deviation observed in increased densities. Since the attacker adds exactly the same value to each beacon, it is expected that the DMV does not perform at all: indeed, this effect can be observed very well in the high density graph (precision remains constant at 0.3, the attacker fraction, for all thresholds). A very similar behavior is shown by the SSC; again, this is expected, since the relative position claimed by the attacker is the same as the ground truth.

Attacker type 4, which transmits a random position from the simulation area (essentially correspond-

ing to a broken GPS), is never detected by the DMV (since the probability that two positions near the same area are chosen is very close to zero). The ART and the SSC have no problems with this attacker, which is quite easy to detect. It can, however, be observed that low ART thresholds result in a low precision. The randomness in attacker type 4 causes worse performance than attacker type 1, as could be expected.

The next attacker, attacker type 8, shows remarkably similar behavior to attacker 2 for the mechanism (again, confirming previous results [141]). However, due to the randomness in this attacker, the attacker is somewhat harder to detect for the ART than before, and cannot be detected at all by the DMV. The SSC, on the other hand, appears to be surprisingly suitable for this attack. From this, we conclude that fusion has the potential to significantly improve performance, which is in line with our previous work [141].

Finally, our last attacker (attacker type 16) is different from the previously discussed attacks, in that it changes the vehicles' messages in a pattern over time (as opposed to manipulation of individual messages independently, as done previously). This is noticeable in the very different detection behavior, in particular of the DMV, since the attacker is essentially converging to a situation where they do not move at all (which the DMV easily detects). However, ART and SSC behavior is comparable to attackers 8 and 1 – as expected: the attack could be seen as a transition from attacker 8 to attacker 1 over some time.

In summary, we can conclude that the ART with a high threshold works well against attackers that transmit erroneous positions (attackers 1 and 4), but has significant difficulties with those that are designed to confuse applications (attackers 2, 8 and 16). Against these malicious cases, the SSC works surprisingly well with lower thresholds, but it is subject to very poor performance against attacker 2. The DMV mechanism works best in dense traffic against attacker 16, and it also does well against attacker 1, but overall its performance is very poor: this mechanism is clearly only suitable to identify very specific attacks. We also note that the SAW does not outperform any other mechanism in any scenario; a future study that includes ghost vehicles (similar to for example, [23]) could show some benefit, but the extremely low precision will require some effort to make this scheme deployable. Finally, note that ART and SSC appear to out-perform each other depending on the scenario (and the configured threshold): these are mechanisms we will focus on for our examination of the dispersion.

8.5 Results: Dispersion of Errors

Now that we have reviewed the detection performance in terms of precision-recall (PR), we examine our new metric based on the Gini index to study how to improve detection performance. Preliminary analysis has shown that the Gini index is not meaningful for small sample sizes as in our low density results, since the population is too small to make meaningful statements (because the sample standard deviation is very large for these results). The data and graphs are available for future analysis, but we caution against drawing conclusions from these for this reason. We therefore focus on a discussion of the high density scenario, which contains enough vehicles to allow for a meaningful analysis of the distribution of error rates. The results are shown in Figure 8.6; as before, each point is the mean of five runs, and the sample standard deviation is indicated. Recall that in our setup, a Gini index closer zero means that the distribution of false positive/negative rates over the vehicles is closer to being equal, without making statements about the actual value.

8.6 Discussion: Dispersion of Errors

In this assessment, we first discuss the false negative dispersion per attacker. This dispersion gives us information about how different the detection performance is depending on the relative position of the attacker and the benign receiver.

For the ART, we observe that the dispersion of false negatives with regards to attacker 1 is very high: this can be explained by the fact that vehicles near the claimed constant false position will not be able to detect it with this mechanism. A similar effect can be observed for attacker 4, while for the other attacks, the dispersion only increases when the threshold is very low. This reflects the increasing recall discussed in the previous section, but remember that the precision also decreases significantly here. The SSC has a very low dispersion of errors for attacker 2, but unfortunately this is also the attacker against

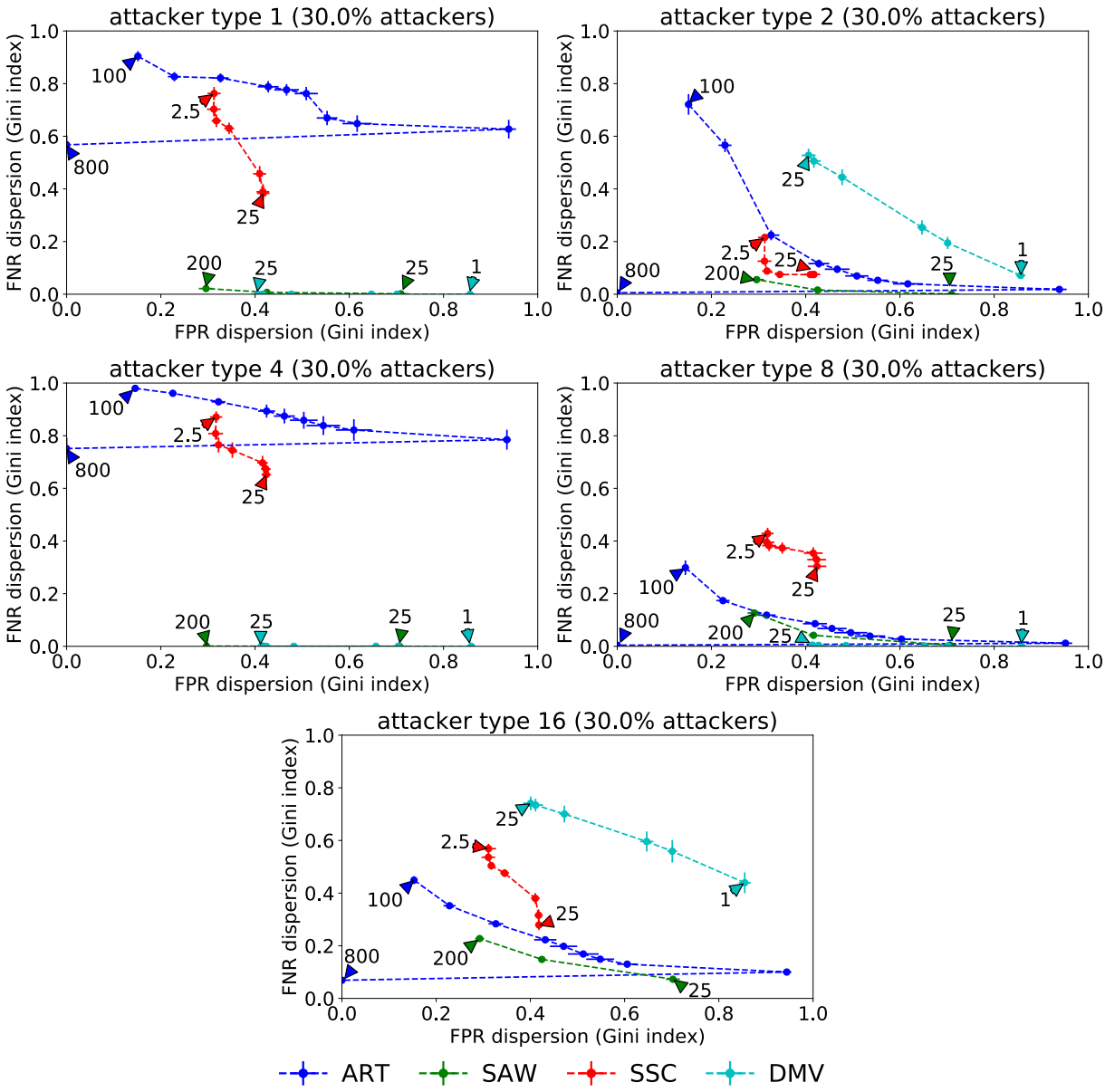


Figure 8.6: Gini indices of FPR and FNR for different attackers.

which its precision is very low. Against attacker 8, the SSC outperforms the ART; the dispersion of errors suggests that this could be a localized effect, meaning that a combination of these mechanisms is likely to be feasible. For the mechanisms that perform very poorly against certain attackers (SAW and DMV), the Gini index shows that their poor performance is not easily fixed: the error dispersion is very close to zero in most cases. The exception is the DMV with regards to attackers 2 and 16: some performance improvement may be achievable by making changes to this detector. Finally, note that for attacker 1, the recall of the DMV is very high, while the Gini index for the false negative rate is very close to zero.

As can be seen in Figure 8.6, the dispersion of false positives for each detector and threshold is the same in every graph. This confirms that the attack configuration does not affect the false positive rate, which is the expected result for detectors that are not subject to bad mouthing attacks. One can observe that the dispersion of false positives for the ART show that for higher thresholds, the number of false positives is significantly skewed over the population; this reflects the intuition that receiving a message from up to 700 meters away is unlikely but not impossible; however, for a threshold of 800, the dispersion is zero. For the SSC, we observe that the threshold is much less relevant to the observed dispersion; this

suggests that the mechanism would need to be changed more fundamentally to flatten the dispersion. A notable case is the DMV: this mechanism has a very high Gini index, meaning that errors only affect some vehicles, and thus is an excellent candidate for fusion with other sources. In this particular setup, where detector assess the reliability of each message from the same source in isolation from other sources, an attack cannot lead to more false positives. Another class of attacks, where an attacker aims to convince a benign vehicle of a false perception of the traffic scenario (e.g., claiming a traffic jam where there is none, by convincing the target that the average speed is much lower than it actually is), this is not necessarily the case. If such an attack is successful, this could cause benign vehicles to make false claims (i.e., misbehave). We refer to this as a *data-driven bad mouthing attack*, a type of attack commonly studied in secure in-network aggregation, where it has been suggested that misbehavior detection could provide benefits [103]. Because our metric studies how benign vehicles incorrectly classify messages as misbehavior, it could also be used to assess the impact of such attacks.

8.7 Summary

In this chapter, we have introduced a new dataset for misbehavior detection in vehicular networks, called *VeReMi*. The purpose of this extensible, publicly available dataset is to provide a basis on which researchers can compare detection results in a wide set of traffic behaviors and attacker implementations. We have additionally shown the application of this dataset to two existing, well-studied detection mechanisms (the ART and the SAW), as well as two simple new detectors (SSC and DMV). We have also provided our results in terms of precision-recall, as well as our newly introduced Gini metric that enables the user to determine which detectors can potentially be improved. Using a combination of these metrics allows developers to have a more holistic view of a detector's assessment, which is information that can also be used in many fusion frameworks. VeReMi will enable other researchers to compete with our detectors, and in the next chapter we will use it to study fusion with the detection mechanisms introduced in this work.

Chapter 9

Fusion & Local Trust

In the previous chapter, we have seen how detector performance is highly dependent on the type of attack, and our results have shown that different types of detectors work differently in different scenarios. This is why Maat uses fusion to combine the outputs of detectors; the goal is to exploit the strengths of each detector to cancel out the weaknesses of the others. Fusion in Maat is based on subjective logic, as discussed in Chapter 5. Subjective logic offers a variety of operators for different scenarios; in this chapter, we include the cumulative belief fusion (CBF), weighted belief fusion (WBF), and consensus and compromise fusion (CCF) operations. We also introduce subjective logic variants of two baseline operations, primarily for comparison: the minimum operation (MIN) and the majority operation (MAJ).

The core contribution of this chapter is a study of the interaction between fusion through subjective logic, the application of transitive trust in detection mechanisms, and the effects of exponential weighted averaging to compensate for detector errors. We analyze and evaluate several different fusion operations (see Section 5.2 for an overview of the fusion operations) using the VeReMi dataset [172] introduced in the previous chapter. In addition, we discuss their applicability of these operations in fusing the outputs of detectors. Additionally, we evaluate whether incorporating past detector outputs for the recent message history using an exponential weighted average (EWA) improves detection results.

The results show that WBF is the best choice overall, where fusionability (i.e., trust in detection mechanisms) controls how close the decisions are to a majority decision (i.e., MAJ). MAJ performance varies strongly depending on how well-suited detectors are to specific attacks, therefore a trade-off needs to be made. Our attempts to improve results through the application of an EWA to compensate for individual detection errors increases recall overall, and for some attacks also increases precision. However, the increased recall should be taken with a grain of salt, since in the dataset we used for analysis, attacks are constant; an attacker could exploit an EWA through carefully crafted periodic message injection.

9.1 Comparison to Related Work

Here we briefly revisit the related work already discussed in Chapter 2 to address some specific differences between Maat and their work in more detail.

Leinmüller & Schmidt et al. [30, 35] have proposed VEBAS, a behavior validation framework that generates positive and negative ratings from different detectors that validate a specific property of a message. Positive ratings are generated for behavior that confirms the sender is a real vehicle[1], while negative ratings are generated for behavior that indicates maliciousness. The ratings from the same detector are combined using an exponential weighted average, and the resulting outputs are combined by summation. Using summation as the fusion operation works, because all the detectors are designed and tuned exactly to work together in this setting. Changing the behavior of one detector thus affects the impact of other detectors on the fused result. The result is then shared with other vehicles as a recommendation, and recommendations from other vehicles are fused to make final decisions about messages. However, the authors later point out that final decisions should instead be made based on the local sum (i.e., the fusion of aged local information), where the trustworthiness threshold is set by an application [30, 35].

[1]The authors assume that malicious vehicles do not exist as such, i.e., they assume an attacker extracts key material and use it outside of a vehicle.

One of the main disadvantages of VEBAS and related works is that their performance relies on the correct behavior of *all* detectors, and the potential impact of errors in some can greatly affect performance. Attacks that cannot be detected by most detectors vehicles are equipped with are thus very unlikely to be detected. Essentially the authors are proposing something like a majority decision: if most detectors accept the message, it is assumed to be legitimate. For example, if an attack affects the acceleration field of a message, while most detectors focus on position, the attacker message is likely to be accepted as legitimate. In addition, the exponential weighted averaging and the fact that a lot of their proposed detectors take a significant amount of time to produce output makes this setup less suitable for immediate response. In our work, we specifically focus on this problem of fast response, and ask: what performance can be achieved using only message-level decisions? We also investigate other fusion operations that can be used for better, more nuanced decision making.

Another significant framework is the work by Stübing et al. [82], whose design targets a specific application: movement validation. This framework consists of several plausibility checks, similar to the detectors in VEBAS, followed by an elaborate tracking scheme inspired by safety applications. This tracking is based on a Kalman filter, which is used to determine the validity of the newly received position information. The decision is made based on the deviation from the predicted position, which leads to obvious problems when vehicles perform maneuvers such as overtaking; for this purpose, a maneuver recognition was added.

One of the most significant disadvantages of the work by Stübing et al. is that their framework was not designed to integrate new detection mechanisms or payloads. Due to the hierarchical nature of their design, the framework is very inflexible, and changes to a single detector or connection require changes to many of the framework components. In addition, even if flexibility would be increased, their hierarchical design of plausibility requires that *all* possible legitimate behavior is accepted. False positives in these checks would cause important messages to be discarded before even being fed into the Kalman filter. In other words, the false positive rate should be extremely close to zero to avoid classifying many vehicles as malicious. However, a low false positive rate also necessarily implies that the false negative rate is higher, since there are significantly more values for the attacker to choose that are accepted by their framework. In our work, we take a more nuanced view, and examine how fusion of these different checks can be performed in a way that detects more attacks.

Raya et al. [33] proposed a number of approaches for data-centric trust. The authors describe an abstract framework, which considers various aspects such as node properties (e.g., node type; is it a police vehicle or a regular vehicle?), security status and a dynamic trust metric. The dynamic trust metric is a function that assigns a trust to each node based on its task within the network. The different factors are combined into trust levels per vehicle, which together with the message are passed to a decision logic to make the final decision. In this specific paper, Raya et al. [33] investigate various options for the decision logic. The studied options in their work include majority voting, weighted voting, Bayesian inference (BI) and Dempster-Shafer theory (DST) as potential candidates. Their discussion points out that in cases with high uncertainty, Dempster-Shafer (DST) performs better than other methods, while Bayesian inference is most resilient to false reports (i.e., falsely issued trust between vehicles).

In our work, we also use a decision logic to combine trust from multiple sources; however, we focus on the point that is not discussed in detail by Raya et al. [33]: trust based on the contents of a message. Based on their results, we choose subjective logic as the decision logic, which can be regarded as a more extensive theory that encompasses DST. In their work, the message's contents are a simple binary event (i.e., a true or false accident report), while in our setting, we consider each beacon message as a potentially true or false event, increasing the amount of decisions that are made compared to their work. In addition, we also have a different application scenario: our goal is to analyze the performance of immediate message-level decisions, rather than wait until the last possible moment to react, in order to gather more information. Our goal is to analyze the potential performance associated with such a filtering system; this information can then be applied to improve the overall system performance in a trust setting as proposed by Raya et al. [33].

One other area that has recently gained interest throughout the security community is the application of tools from the area of machine learning. Although we feel that machine learning will likely have its place in our field in the long term, we remark that the blind application of machine learning techniques to detect attacks is not a reliable one, as recent research has shown [169]. It is important to at least apply new attacks as part of the validation of the algorithm, which is currently not common practice in our field, regardless of whether machine learning is applied [71, 141, 154]. As the publication of results along

with source code and data set is very uncommon in neither this field, nor that of misbehavior detection, it is difficult to compare detectors with machine learning approaches. Since the detection performance of machine learning models is typically bounded by the input data quality and availability, overfitting might be a significant issue. Therefore, we do not claim that our work performs significantly better than other systems, but instead show how to tune our framework to improve performance. Machine learning could then be used to optimize performance in future work.

9.2 Fusion & Trust Revisited

As discussed in Chapter 5, our system applies a combination of trust transitivity, fusion and exponential weighted averaging in order to decide on the trustworthiness of data based on the received messages. We briefly revisit the setup: the detectors in Maat perform their detection and output statements of trustworthiness based on the internal logic of the detector, as evaluated in the previous chapter. In this chapter, we focus on the remaining parts of the system, in particular the fusionability, fusion and exponential weighted averaging. This is shown schematically in Figure 9.1 and was previously discussed in detail in Section 5.5. We briefly revisit these elements in this section.

9.2.1 Fusionability

We also consider a base fusionability parameter that determines how much trust we have in each detector. This was previously introduced in Section 6.3.3, and we briefly repeat our ideas here. The value for fusionability is chosen equal for every detector in this study. Since detectors have different confidence values, a different choice for the base trust for each detector still provides different outputs. To study the broad effects that high and low trust in detectors has, we choose $0.1, 0.3, 0.5, 0.7, 0.9$ as trust values to analyze. These values were chosen to validate our expectation that very high trust in detectors causes the operations to approximate a majority decision (i.e., MAJ), and that a very low detector trust causes them to approximate a minimum (i.e., MIN). This is because the lower the trust in a detector, the higher the resulting uncertainty that is used for fusion afterwards.

These trust values are converted into opinions by setting the belief to the chosen value v, the disbelief to $1-v$ and subsequently computing the uncertainty-maximized opinion. This ensures that the projection of the opinion is the chosen value. Recall that the fusionability essentially represents the trust in a detector. To combine this trust with the detector output, trust discounting is applied. In other words, the fusionability can be seen as a weight that specifies how important the belief or disbelief of the detector output is. Note that the fusionability could also be determined dynamically to make the system more adaptive; however, adequately designing such a dynamic system requires information about how this transitive trust in detectors affects detection performance.

9.2.2 Exponential Weighted Averaging

The purpose of an exponential weighted average, which we apply *before* fusion (i.e., on the output previously provided by the detector), is to even out "random" classification errors of a detector. Since it is commonly taken for granted that the EWA improves performance in related work (see Chapter 2), we include this behavior in our system as well. In order to apply this properly, we have defined the EWA for subjective logic with the following equation in Section 5.4.2:

$$E(\omega_t^{\mathrm{A}}) = (1-\alpha) \cdot E(\omega_{t-1}^{\mathrm{A}}) + \alpha \cdot E(\omega_{\mathrm{new}}^{A}) \tag{9.1}$$

9.2.3 Fusion Operations

The fusion operations we compare are the cumulative belief fusion (CBF), weighted belief fusion (WBF), the consensus and compromise fusion (CCF), as well as the minimum (MIN) and the majority (MAJ) operations. These were previously introduced in Section 5.2. Recall that the CBF works cumulatively (i.e., detectors are considered independent), while WBF uses confidence-weighting and CCF outputs a compromise between its inputs. MAJ and MIN represent the majority and minimum trust approach respectively. The majority decision assumes that all detectors have a similar performance regardless of the attack or lack there-of. The minimum assumes that all detectors are quite competent at detecting

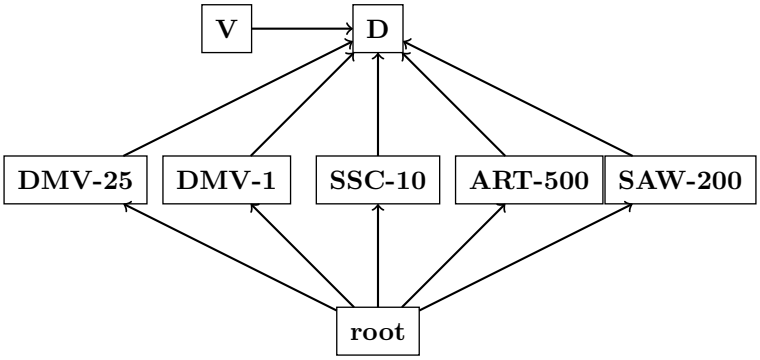


Figure 9.1: A world model with a single data item D and a single other vehicle V.

attacks, but they detect different types of attacks, and thus assumes whenever a detector is triggered, an attack must be occurring.

9.2.4 Summary of Evaluated Components

The work from Chapter 8 has shown how detectors behave when evaluated independently with different thresholds. In this chapter, we focus on all the surrounding components that Maat adds into the system. A full discussion of Maat's complete fusion process was provided in Section 5.5, which also includes an extensive flowchart that shows how each component is related. We do not repeat this discussion here, but rather list the evaluated components that are the subject of this study:

- exponential weighted averaging (EWA) for classification error correction
- fusionability: the effect of trust in detectors
- fusion operations: how well these compensate individual failures of detectors

As discussed previously, we use EWA for each detector to incorporate past opinions on the message history before fusing the EWA adapted results. Thus, a choice must be made which value is chosen to represent the output in opinion space to fuse in the next step. In summary, we provide results for the behavior of the *fusionability*, which models the trust in detectors, as well as the chosen fusion operation, and finally the use of the EWA.

9.3 Evaluation Methodology

The previous chapter has introduced a dataset for the evaluation of misbehavior detection, named the VeReMi dataset [172]. This dataset, generated from state of the art simulation models that cover a wide variety of different scenarios, is intended to be a baseline comparison for misbehavior detection mechanisms. In that work, we also present a preliminary set of results for several detection mechanisms, for which all of the corresponding source code is publicly available. We adopt their results and examine the impact of fusion between these proposed mechanisms. Therefore, we briefly revisit the proposed detection mechanisms, select threshold values for each, and examine the impact of different fusion operations on this basis. The dataset contains a number of different attacks, some of which are designed to circumvent the detection mechanisms; this setting is ideal for our fusion experiments, as it exactly matches the setting described in the previous section.

9.3.1 Attacks

As discussed in Chapter 7, the attacks in the VeReMi dataset are variants of data injection attacks, executed independent of one another. The malicious behavior of the attacker is consistently executed throughout the simulation, and the baseline is therefore that all messages sent by the attacker should be classified as malicious. The attacks in the dataset are a constant erroneous position (attacker 1), a constant offset to the real position (attacker 2), a random position (attacker 4), a random offset to the real position (attacker 8) and a random stop (attacker 16) [172]. For the random attacks, randomness

is newly applied for each message (i.e., attacker 8 may send an offset of 5 meters, followed by an offset of 200). In general, it is possible that some of these attack patterns may cause certain messages to correspond almost completely to a legitimate message (e.g., the random offset of attacker 8 is zero). It is conceptually possible to limit which malicious messages should be classified as malicious; however, this limit is necessarily arbitrary. Therefore, we instead grant that there will be some inaccuracies in our assessments and require the detection of *all* attacker messages for perfect precision and recall. This is analogous to what was done in the previous chapter.

9.3.2 Detection Mechanisms

In the work introducing VeReMi [172], we used four mechanisms: a simple speed check, a distance moved verifier, an acceptance range threshold, and a sudden appearance warning. These mechanisms were previously discussed in Chapter 6. We also apply these mechanisms here, because they satisfy our requirement that decisions be made immediately on the message level (rather than waiting for multiple messages, as previous work has proposed). Each of these mechanisms has a threshold parameter that specifies at which point the observed data is considered malicious. These parameters were previously studied in Chapter 8, and here we only briefly re-introduce the detectors and the parameters we use.

The simple speed check (SSC) is a detector newly proposed in [172], and represents a simplification of the Kalman filter proposed by Stübing et al. [82], as the necessary parameters for the Kalman filter are not provided by Stübing et al. [82]. Our attempts at reproducing their results failed, and thus we elected to design the SSC, which for every given message examines the difference between the current claimed speed and the speed computed from the time and position differences from the previous beacon message. The threshold decides what difference between these messages is considered acceptable. In our work, we use the value $\tau = 10\ m/s$ as the threshold for maliciousness, which performed particularly well against the attacker that performs a sudden stop (attacker 16 as discussed in [172]). As this attack is difficult to detect by other mechanisms, we choose a threshold for SSC that detects this attacker well. The final opinion to represent this output has a minimum uncertainty of 0.1, and a disbelief of $\frac{\Delta v}{\tau}$ (set to 0.9 if the fraction too large); the remaining mass is assigned to belief.

The distance moved verifier (DMV) is also a detector that was proposed in [172], which detects attacks where a stationary position is claimed, as previously discussed by [56]. The threshold of this verifier is set to the minimum movement required between beacons. This detector was clearly inspired by VEBAS' minimum distance moved, which performs a similar task, but takes a longer list of messages to output a result. Because it was observed that this detector works very well at different thresholds depending on the specific attacker, we use the extreme thresholds 1 and 25 as thresholds in this work. Thus, we essentially have two instances of this detector, detecting attacks either with high recall (threshold 25) or with high precision (threshold 1). The output opinion of this detector is 0.5 belief or disbelief, depending on the threshold, and 0.5 uncertainty.

The acceptance range threshold (ART) tests whether the received message is in the reception range of the receiver. This detector has been relatively well-studied [30, 141], from which the most important conclusion is that this threshold is very sensitive to the chosen channel model. Because reception range varies greatly between open areas (e.g., highways) and urban areas (where buildings interfere with transmission), finding an appropriate threshold that still detects the majority of malicious messages has proven to be difficult. In this work, we use 500 as the threshold, which has previously been shown to be a good threshold [141], and we [172] have previously shown that this value provides a balance between precision and recall. The opinion used to represent the output is chosen the same way as in previous work [141].

The sudden appearance warning (SAW) tests whether new vehicles suddenly claim to appear close to the receiver, indicative of an attack that aims to trigger a breaking response from the receiver. This detector has very poor detection performance in terms of false negative rate, as it accepts all messages that do not trigger this warning. However, detecting this specific attack may be invaluable, and therefore this detector represents an example of the highly specific detectors discussed in Section 9.2. Since the best results were always achieved for threshold values of 200, we adopt this threshold value, while the opinion is chosen analogously to ART.

9.4 Fusion: Results & Discussion

We now discuss our results for the different fusion operations, introduced in Section 9.2. As discussed there, we expect that the results produced by the MIN and MAJ operations depend strongly on the detectors included in their input. In these results, several detectors of varying quality are included, thus illustrating the effects of such variations. Our expectation regarding the specific fusion operations is that they fill the intermediate space between the two extremes, MIN and MAJ.

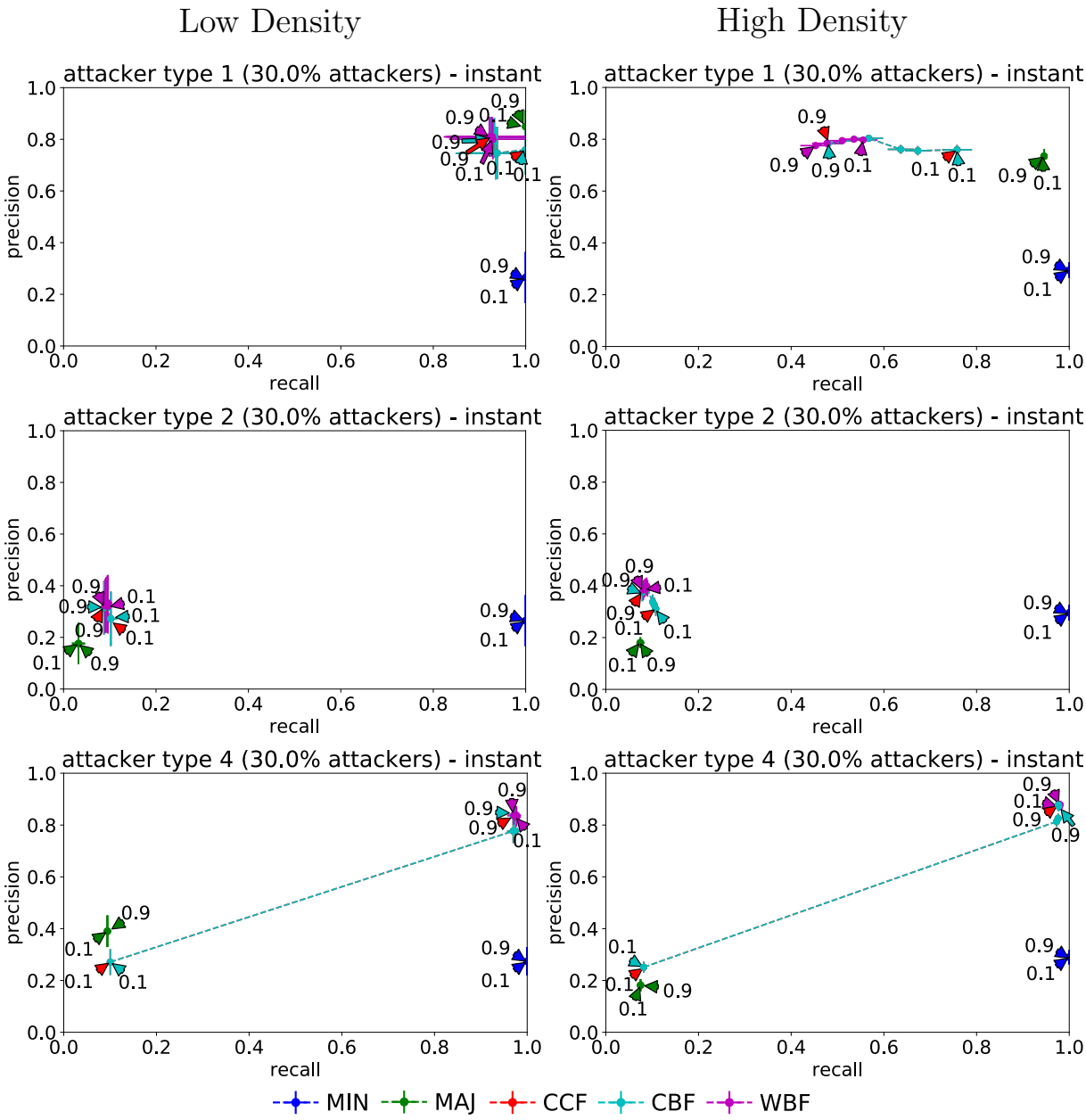


Figure 9.2: Precision-recall graphs for low densities (left) and high densities (right) for attackers 1, 2, and 4, without EWA. The numerical labeling refers to the projection of the trust in the opinion.

The results of our detection runs are shown in Figures 9.2 and 9.3 in the form of precision-recall (PR) graphs. We here show the results for high attacker densities (30% attackers) only; similar results (when compensating for prevalence) are achieved for the other parts of the data set, with 20% and 10% attackers. Each plot includes both the five fusion operations and the projected value of the fusionability parameter, which was evaluated with the values $0.1, 0.3, 0.5, 0.7, 0.9$; the arrows in the figures indicate

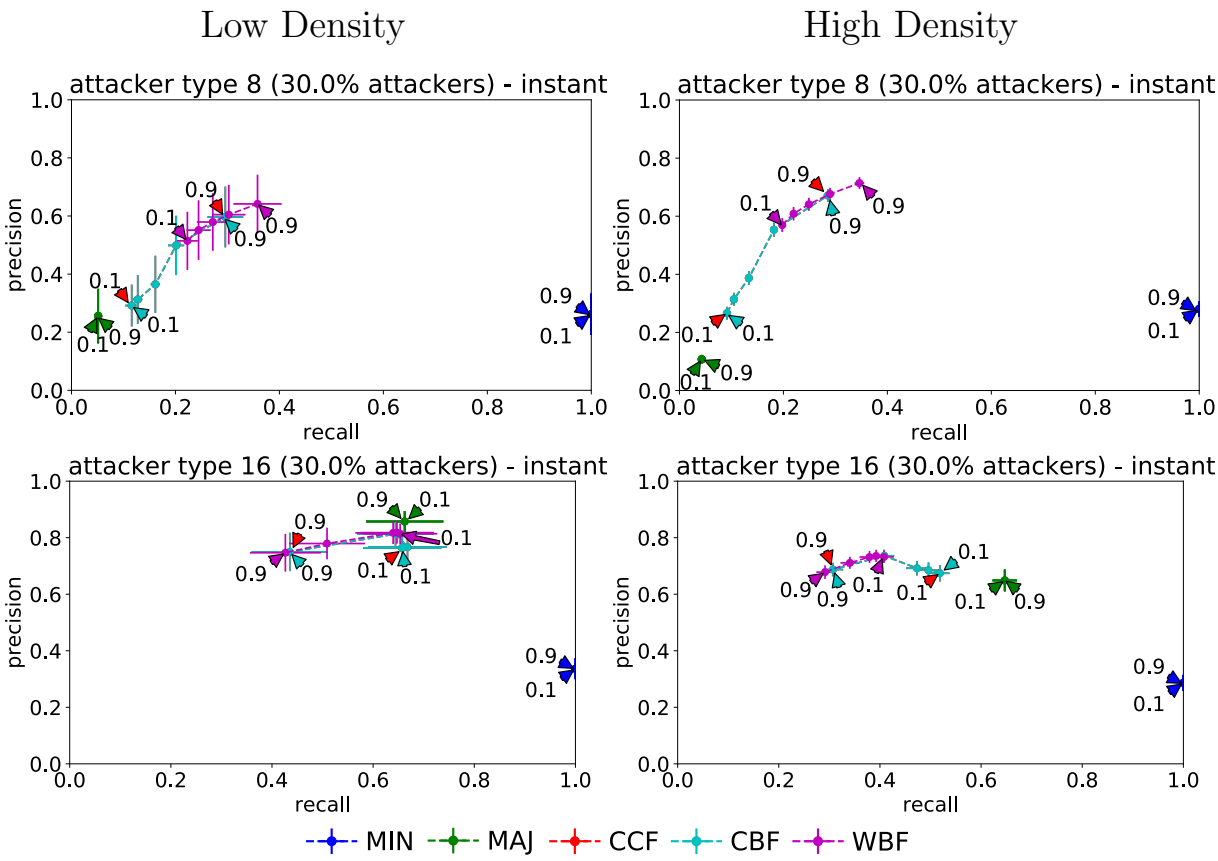


Figure 9.3: Precision-recall graphs for low densities (left) and high densities (right) for attackers 8 and 16, without EWA. The numerical labeling refers to the projection of the trust in the opinion.

the minimum and maximum, while a line connects the other values. The motivation for these values is provided in Section 9.2.

The first important observation is that MIN always has maximum recall, while precision varies around the prevalence (i.e., ≈ 0.3). This means that MIN almost always classifies as an attack; it essentially takes the best of each detector for the detection of attacks, resulting in practically no false negatives. However, it is also very imprecise, since as soon as any detector has a false positive, or some indication of disbelief, the MIN operation combines this into a false positive. This re-enforces the need for fusion – in almost all situations, some detectors have false positives; here, SAW is the primary cause, as its performance is very poor (see Chapter 8). Recall that this is by design, i.e., SAW is integrated in Maat to detect a very specific attack, and output complete uncertainty otherwise. This causes MIN to perform poorly. On the other hand, MAJ works best or worst depending on the specific attack; this is due to the fact that for MAJ to detect an attack, the majority of the detectors must agree. Since not all detectors detect all attacks, the attack type acts as the most important factor, which clearly is not a desirable situation. An attack that triggers one detector with high certainty will not trigger MAJ, thus MAJ performs poorly for attackers 4 and 8 (which are the hardest to detect). On the other hand, attacker 16 is more likely to be detected using MAJ than by using one of the other fusion operations. These results are in line with the predictions we made in advance, and they show in particular that the base line fusion operations are not desirable.

Having discussed the base line operations, we now examine the results for the subjective logic operations. The first observation is that CCF and CBF perform the same in all graphs; after a discussion with Audun Jøsang, author of subjective logic, we verified that this is not due to a programming error, but rather it is an artifact of how CCF is defined for binary domains. In other words, this result caused by the underlying commonality between these detectors; although they differ in output, the decisions generated by these detectors turn out to be equivalent. Looking back at the theory, we deduce that the

cause is that both operations treat uncertainty in a similar way. Any differences that could arise in the general case do not apply to our situation, because those differences are associated with the treatment of composite values in the domain. In binomial domains, the only composite value is the uncertainty, and therefore both operations predict the same outcome. Since the CCF operation is computationally expensive for large input sets, we recommend the use of CBF over CCF.

What remains is a comparison between CBF and WBF. For the first three attackers, as shown in Figure 9.2, the performance of CBF and WBF is comparable in most cases, with the exception of attacker 1 in a high density setting, and attacker 4. Against attacker 1 in a high density setting, fusion performance is quite low; this is caused by the fact that the DMV and SSC detectors all perform worse in a high density scenario with stop-and-go traffic. Since the vehicles pass the position claimed by attacker 1, the ART is not sufficiently effective to counteract the reduced performance of the other detectors, and the WBF is affected slightly more by this issue. CBF with low detector trust is affected the least by this, since the uncertainty before fusion is highest here. However, CBF with low detector trust has very poor performance against attacker 4 in both low and high density settings.

A similar observation to that of attacker 1 can be made regarding attacker 16, where the low detector trust CBF also performs well. Note that this attack, once it is fully active, is exactly the same as attacker 1, except that the constant position is a position that was true at some point. However, the attacks that show random behavior (i.e., attacker 2 and 8) show that WBF outperforms the other fusion algorithms, particularly with high detector trust. This suggests that despite not modifying the detectors for the attacks we applied, WBF can perform well against these attacks. Note also that by definition, CBF will reduce the fused uncertainty as the amount of detectors goes up, which suggests that as we implement more detectors for different attacks, a negative performance impact can be expected[2]. Based on these results, we recommend that WBF be applied in the fusion process, as it shows acceptable performance against attacks that are not previously known.

In summary, detecting these five attacks is surprisingly difficult; we find that generalized over all attacks, the CBF and WBF operations appears to work best. MIN detects practically all attacks, but has a very low precision, thus implying a huge FPR. MAJ detects some attacks well, but other attacks poorly; this indicates that the configured set of detectors significantly influences the result. Adding a detector could completely change MAJ results, and would also affect CBF performance significantly. Thus, we recommend WBF. Fusion with subjective logic lacks the disadvantages of MIN and MAJ, but introduce their own; the overall performance is not very good on a message basis, despite being better than individual detectors.

Attempting to compensate for these errors through the use of detector trust does not solve this problem immediately; although the choice of this value meaningfully influences the output (see e.g., attackers 4, 8 and 16), there is no clear-cut best value. Although this is not particularly surprising, what is notable is that for attacker 1 in high density scenarios and for attacker 16, a very low trust in detectors performs well, while on the other hand, for attacker 4 and 8, the exact opposite is the case. When comparing with the MAJ detector, it can be seen that in every setting, lower trust in detectors tends more towards a majority decision. This is useful information for future work: the trust can be used to configure how strongly the system is to be influenced by the majority of detectors.

In general, we can conclude that for our intended design, where many different, highly specific detectors are designed to detect the presence of specific attacks, a choice with low trust seems the best choice. However, in particular attacks 2 and 8 show that significant improvements should be made to the detectors themselves, as fusion can only correct for so much. Fusion can perform the desired task of raising the performance to some extent, but not beyond the capabilities of individual detectors. Previous work [35, 88] has suggested an EWA may be a simple mechanism that corrects errors in detection; thus as a simple improvement of the output of each mechanism, we apply EWA and study the resulting fused output.

[2]This is because CBF is designed to accumulate evidence, as the name suggests. The issue is that this only works if the evidence suggests the same thing, yet evidence from detectors that see different attacks will provide conflicting evidence (of different attacks). It is well-known [144] that CBF does not deal well with this type of conflict. Thus, although it seems obvious that CBF will not work well in a system with many different detectors and attacks, it should still be confirmed by experiment. It may be possible to provide design guidelines for detectors that can account for this weakness of CBF.

9.5 Exponential Weighted Averaging: Results & Discussion

We experiment with different values of EWA: $\alpha = 0.5$ (equal weight to new and old information) and $\alpha = 0.1$ (old information is weighted significantly stronger than new information). We expect EWA to improve results by a lot if the detectors' errors are random. For false positives, we expect EWA to be particularly beneficial, since this not related to malicious behavior. For these experiments, we also expect increased recall, since the attacker model is quite simple. More precisely, all attacks are constant; if we allow intermittent attacks (where the attacker changes its behavior), recall will likely drop significantly. Choosing a small α is likely to be vulnerable to such intermittent attacks. We thus study a small choice for α and the equal balance between new and old information to determine the effectiveness of this EWA, with the goal to inform future work that analyzes the EWA under these more difficult conditions.

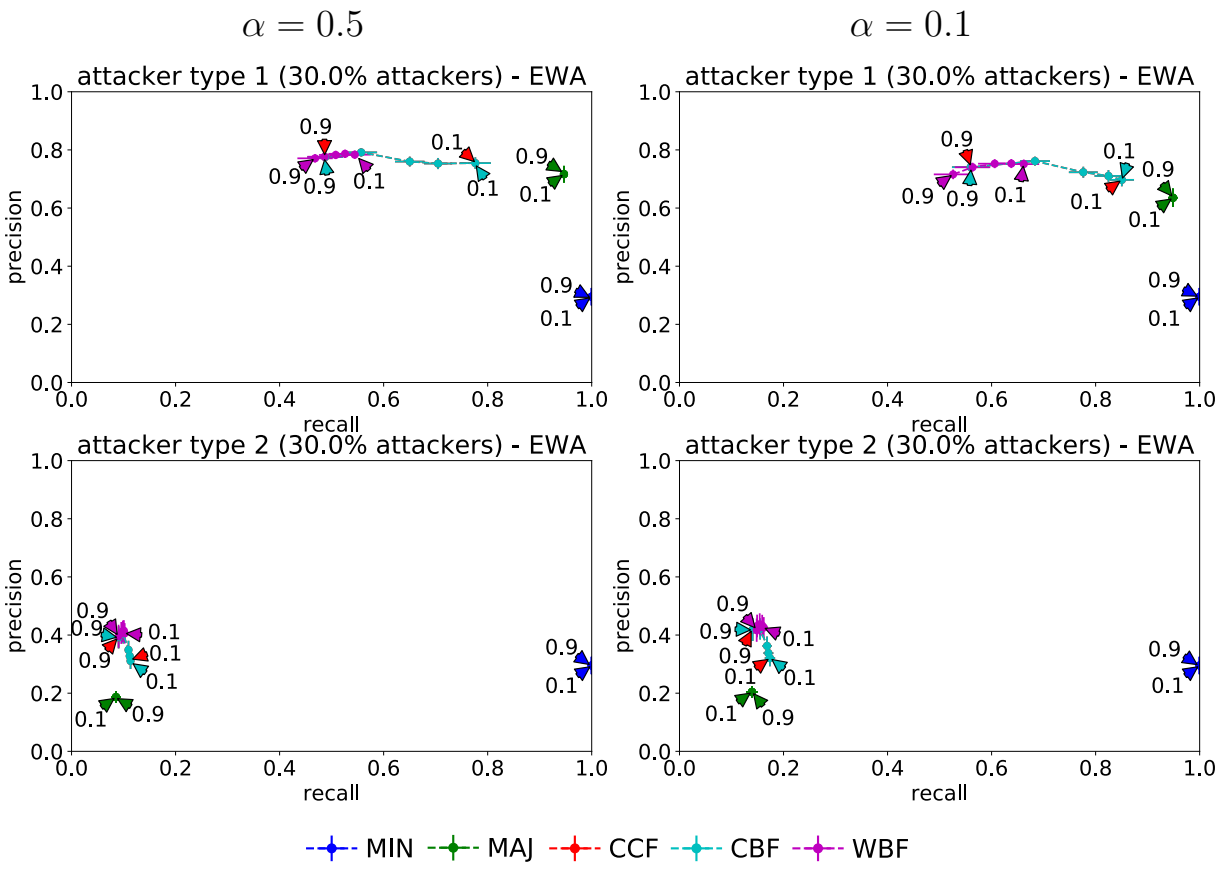


Figure 9.4: Precision-Recall graphs with respect to attackers 1, and 2 with EWA: $\alpha = 0.5$ to the left, $\alpha = 0.1$ to the right.

In Figures 9.4 and 9.5, we present the PR graphs for the high density cases from Figures 9.2 and 9.3, now applying an EWA to each detector's results, followed the fusion operations, as discussed in Section 9.2. The graphs for low densities show similar behavior, and can be accessed in the same data repository. As can be observed by comparing these results with the results from the previous section, the improvements offered by the EWA are quite limited overall. However, it can also be observed that everywhere, the recall improves significantly, while for precision it depends on the attack what improvement is observed (see e.g., attacker 8, where both precision and recall improve, and attacker 16, where recall comes at the cost of precision). Therefore, contrary to our expectation, we can make no universal statements recommending the use of EWA, despite the advantageous setting. The setting is particularly advantageous, because the attacker maliciously modifies all packets they transmit; traditionally, the weakness of the EWA is that it is vulnerable to periodic attack behavior. For example, an attacker might only manipulate one in three messages; this would cause the EWA-based system to accept all messages as legitimate. For this reason, we recommend avoiding the EWA in the form presented here.

However, future work on a specialized EWA variant that takes advantage of the evidence implied by a subjective logic opinion may compensate for such attacks. We thus suggest that such techniques should be used and tested against reputation style attacks, and to avoid naive EWA for misbehavior detection.

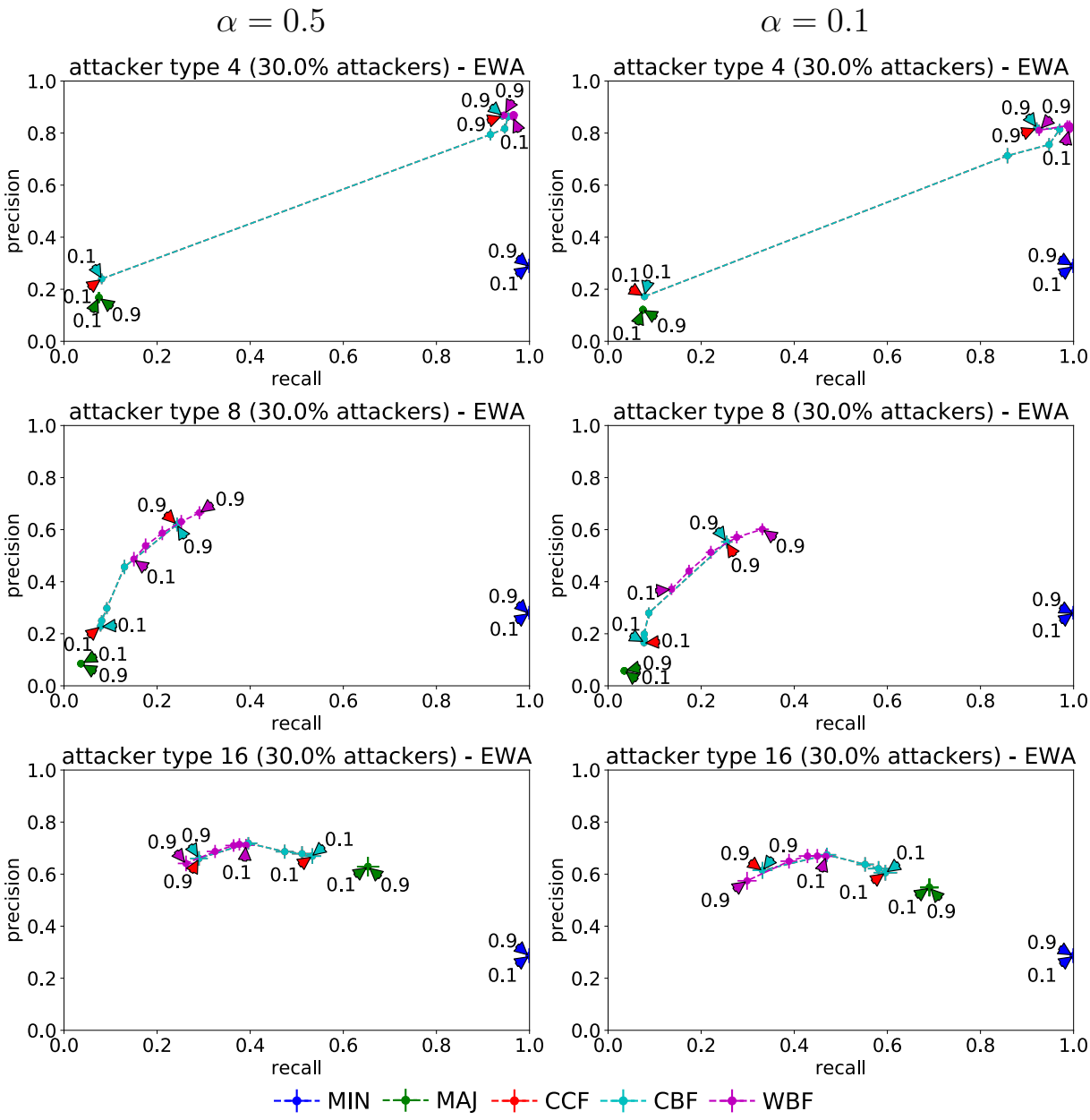


Figure 9.5: Precision-Recall graphs with respect to attackers 4, 8 and 16, with EWA: $\alpha = 0.5$ to the left, $\alpha = 0.1$ to the right.

9.6 Conclusion of Fusion Evaluation

We have discussed different fusion operations for the fusion of misbehavior detection mechanisms. Of the many available operations, WBF is the most balanced over all the results, while MIN catches all attackers but has extremely low precision. As discussed previously, it is also the operation that is least susceptible to errors caused by the addition of new detectors to the fusion process. The primary goal of this study was to determine how well fusion can compensate for the errors produced by some

detectors. WBF adequately performs this task if the attacks are unknown; CBF and MAJ perform better if the detectors are designed specifically to detect attacks present in the dataset. Thus, for misbehavior detection in the short term, WBF is the best candidate in our opinion. For revocation or punishment of detected misbehavior, MAJ could be used along with a human agent who judges the results. Such a forensic analysis may be realized through Maat using the reporting process outlined in Section 4.3. We recommend that an EWA be avoided in all situations. Finally, we recommend that fusionability be configured to a high value, unless there is evidence that a specific detector performs poorly in a specific scenario. Finding and correctly recognizing such specific scenarios is out of the scope of this study. Because Maat preserves all the history, a potential solution could be to apply machine learning techniques to this problem.

9.7 Overall Evaluation Conclusion

Combined with the previous chapter, this concludes our evaluation of Maat. We have analyzed the entire fusion process that occurs after a query for a message is made, as outlined in Section 5.5. In our evaluation, we focused on immediate evaluation of received messages, which is the most difficult classification task available for detectors. Success in this task enables a rapid response in the form of reaction and recovery, which is not available as a next step in the approach taken by others. In addition, we decided to primarily target fusion to ensure reasonable performance regardless of the quality of individual detectors. Although this does not necessarily guarantee the best overall performance, it makes sure that fusion can deal with the inevitable poor performance that some deployed detectors will have.

Many more experiments can be performed based on our results to further improve detection rates and to design better detectors. The primary goal of our evaluation chapters was not to exhaustively and conclusively show that Maat performs best in every regard, but rather to improve the overall understanding of misbehavior detection. Thus, we focused on a nuanced discussion of new metrics and different options of fusion, rather than claiming to present a final solution for the problem. In Chapter 8, we discussed how our dataset was generated, and we demonstrated the performance of individual detection mechanisms introduced in Chapter 6. Our evaluation clearly shows that no single detector we presented can detect all the attacks on the C-ITS.

The objective of this chapter was to compare different fusion operations that were described in Chapter 5. For fusion operations, a similar statement applies: there is no perfect fusion operation that will always perform better than all other operations. By evaluating different attacks in different traffic densities, we were able to determine the dependencies between the parameters of our fusion process and the resulting performance per attacker. In real systems, attacks are unpredictable, therefore we discussed how Maat may be tuned to avoid being dependent on the type of attack, and rather have a good performance across all attacks. We have also learned here that applying an EWA to detection results appears to cause poorer performance almost universally, despite the fact that our attacks are continuous and not designed to avoid detection by an EWA. Therefore, we strongly advise against its usage in the future.

Part IV

Implications & Conclusion

Chapter 10

Conclusion

In this thesis, we have provided a number of contributions in the area of misbehavior detection in cooperative intelligent transport systems. Cooperative intelligent transport systems are networks of cooperating vehicles, which may be partially or even fully autonomous. The communication in these networks aims to provide additional safety and comfort features for drivers and passengers. Wide-scale adoption of cooperative intelligent transport systems will likely enable city-wide applications such as increased fuel efficiency and automated traffic lights. The primary objective of misbehavior detection in this domain is the efficient and timely detection of attacks by malicious or faulty participants in the system.

In our work, we have described a world modeling and fusion approach to enable the flexible integration of individual detection mechanisms. This enables the detection of both existing and new attacks by exploiting knowledge of the underlying physical processes. The world model we design provides a basis on which detection can be performed in a coherent manner, while remaining independent from the specifics of the communication stack. Through our application of subjective logic, detector output can be converted and fused into a single decision, either when receiving messages or when data is queried from an application. Subjective logic provides a number of different fusion operations, which enable its flexibility. As part of our work, we describe multi-source extensions of the fusion operators that are non-commutative, enabling us to transfer this flexibility to a scenario with multiple sources. Besides enabling our use of subjective logic in a consistent manner, this contribution enables multi-source fusion in any setting where subjective logic is applied.

Our contributions are prototypically implemented in the Maat detection framework, which is an open source system that demonstrates the efficacy of our approach. As part of this implementation, we also provide a library of detection mechanisms for which no public implementations were previously available. To demonstrate the performance of our fusion approach, we also provide an extensive discussion of methodological issues that are prevalent in our field. In particular, we show that existing work is not replicable or reproducible, and provide a detailed exposition of potential solutions. One of the issues is that there is no public dataset that can be used for the comparison of misbehavior detection mechanisms. To resolve this issue, we design a public and repeatable setup for simulation experiments that is modular enough to allow extensions with new attacks. An initial dataset, called VeReMi, with five types of attacks is published and designed to be a baseline for any evaluation of misbehavior detection work. VeReMi includes a wide variety of traffic scenarios and densities, making it suitable for the assessment of detection performance of widely deployed cooperative intelligent transport systems.

We use VeReMi to compare the different detection mechanisms we propose, as well as the overall performance of fusion operations. Our evaluation places much stricter requirements on the detection than previous work, requiring that every attacker-generated message be classified as malicious by every vehicle to achieve a true positive rate of 1. Previous work has often focused on detecting the presence of one or more attacker by every vehicle. In this stricter evaluation scenario, we show that individual detectors perform significantly worse than previously reported. We also show that Maat can outperform both the individual detectors and naive fusion proposals based on earlier work in most scenarios, as evidenced by the results in Chapter 9.

In addition to this comparison of raw detection performance, we introduce novel metrics for evaluation of the suitability of detectors. First, we introduce a novel application of the Gini Index to determine how

the detection performance of individual vehicles relates to the overall detection performance. The Gini index measures dispersion in a distribution; we apply this measure to estimate how false positive and false negative rates are distributed over vehicles. This metric can be used to determine the suitability of a specific detector in a specific traffic situation, which can then be applied by an expert to decide on the relative weighting of the detector outputs. We also discuss a spectrum of application-specific metrics related to the cooperative adaptive cruise control application. We demonstrate the importance of considering application-specific metrics in addition to detection rates, noting that misbehavior affects every application differently.

10.1 Research Questions

We now revisit the research questions introduced in Section 1.3, which were divided into three blocks. The three blocks ask questions about detection performance, functionality and methodology, respectively. Each block has several sub-questions; in the following, we put these questions in the context of our results. In the next and final chapter, we discuss the limitations of Maat and observe how these questions have changed and led to new open research challenges that can be addressed in the future.

In the first block, which focuses on detection performance, we asked Research Question 1:

> How can performance of misbehavior detection in cooperative intelligent transport systems be improved to support a vehicle's autonomous decision making?

This question addressed two key points of novelty contained within this thesis; the context of C-ITSs and the improved detection performance. The C-ITS context of our work, specifically the trend towards its' usage in autonomous driving, implies that "eventual detection" of an attack is no longer sufficient. This is the case even if it is within a time frame of seconds or minutes; we rather require immediate identification of misbehavior. Within this context, we derived three sub-questions that ask, respectively, whether detectors can independently deal with noisy information, how data fusion can contribute, and how fusion results can be improved by modifications of detectors. In regards to noisy and uncertain information, we have shown in Chapter 8 that it is unlikely that one can design a single detector that can both deal with a broad variety of attacks and deal with a wide variation of traffic scenarios. In response to the second sub-question, we designed Maat to apply data fusion with subjective logic, in order to arrive at significantly improved performance beyond baseline fusion operations. Finally, we describe a variety of means to improve performance by further weighing detection results (*fusionability*) and propose a model for improving detector performance beyond hard thresholds. In summary, we have shown that a variety of existing and new tools can improve detection performance, despite the more stringent requirements.

Our second block aims at data and detection result management, guided by Research Question 2:

> How can data be stored, managed, and exchanged to support improved detection, inclusion of new detection mechanisms, and validation of detection?

This question covers the different design goals that we aimed for in our development of Maat. We further sub-divided this into sub-questions concerning how data is stored, how detection results can be fused, and how Maat can be used for revocation. The Maat world model addresses the first question, providing a generic storage framework for the data arriving at a vehicle and an intuitive representation for detector output through directed edges. By labeling edges with subjective logic opinions, we enable a fusion process based on subjective logic, through which detection mechanisms with appropriately modeled confidence can be weighed without further intervention. However, our results show that multiple instances of detectors that use different thresholds perform better than a single instance with a single thresholds. To be able to take advantage of this, we introduce a weighing factor to represent the importance of each detectors output, allowing Maat to use multiple copies of the same detector with different thresholds to exploit the advantages of instances. The Maat world model also provides an intuitive way to reproduce detection results by sending a sub-graph to other entities, which can validate and use these results. This is an advantage over existing approaches, which required trust in the reporting entities in order to perform this task. Maat thus provides a reliable means to perform global revocation. In the future, techniques from secure aggregation may be applied to reduce the large bandwidth overhead required in this approach, enabling wide-scale deployment of Maat for revocation purposes.

Our third block of questions addressed methodological concerns with respect to earlier work, which are raised by the new context in which we asked the other research questions. The formulation of Research Question 3 was:

> How can we support the scientific development and analysis of misbehavior detection in cooperative intelligent transport systems?

This was divided into two sub-questions, addressing the necessary methodological improvements, and the design of new metrics. To answer questions on methodology, we provided a detailed review of methodological practices in this thesis, followed by a description of our own methodology and its advantages. As part of this new methodology, we also introduce a variety of new metrics that in part expose weaknesses in existing work. We use these metrics to demonstrate the utility of fusion, but also describe how the metrics can be used to direct future research by showing where attacks expose the system to misbehavior. These ideas are valuable not just for the design of Maat, or misbehavior detection in C-ITSs, but they also provide insights into developments in cyber-physical systems. We reviewed how misbehavior detection in C-ITSs can be applied to cyber-physical systems, describing the commonalities and differences between the two. Because our work is publicly available and more methodologically rigorous than previous work, we expect these contributions to provide a lasting impact on the field of misbehavior detection as a whole.

10.2 Recurring Patterns in C-ITS

Having discussed the immediate results described in the thesis, we now consider the wider field of misbehavior detection in C-ITSs, as previously discussed in our survey [170]. We extend this discussion to explain how Maat connects to these developments in the wider field, supporting future research in those areas and ensuring integration with our results. Maat can also support future research by addressing some of the open issues raised in our survey; we discuss these in detail in Section 11.3.

Physical models A number of mechanisms use physical models to detect misbehavior. These models allow a detection mechanism to incorporate specific knowledge about the driving behavior of vehicles. The complexity of models ranges from very simple (e.g., cars cannot drive faster than 500 km/h [30], two cars cannot occupy the exact same position [45]) to very complex models (e.g., analysis of acceleration and deceleration behavior [167], movement prediction including turn probabilities [108], correlation of vehicle positions to street maps). These models may also include the sensors of the observing vehicle. Due to their effectiveness, as well as the fact that these mechanisms are able to validate incoming data rather than relying on node trust, physical models have long been considered essential. An important question in this context is the appropriate choice and combination of models to cover a variety of situations a vehicle may encounter.

One of the primary advantages of Maat is that the inclusion of multiple physical models describing a specific phenomenon can easily be done by integrating these as detectors. By using subjective logic in a positive formulation (i.e., a high belief that *the data corresponds to real-world behavior*), the accuracy of the physical model can also be included. This allows a representation in which physical models can be designed independently and added on the fly (e.g., through an over-the-air update), without requiring complete reconfiguration of the system. Maat's design also enables the standardization of models, enabling the exchange of detection results between vehicles and the validation of such results at a global level.

Centralized detection Early works often perform the whole process of misbehavior detection locally and only send reports about local decisions to the back-end to perform pseudonym resolution and revocation. In these works, the back-end does not perform any detection task; rather, it just verifies received reports for correctness and checks for false accusations. However, performing detection exclusively in a local fashion prevents the detection mechanism from detecting larger clusters of distributed misbehavior. For instance, an attacker could repeatedly mount attacks at different locations, which might be detected locally, but only in correlation show the full extend of the attacks' severity. Hence, newer schemes include the back-end more and more in the actual detection work. Notably [89] and [86] employ such centralized detection approaches. A challenge to solve in this context is the level of event reporting. In order to

allow the back-end to detect more attacks than possible locally, it may be necessary to report more data to the back-end, including even mildly suspicious behavior, to provide a large enough data basis for centralized detection.

As discussed previously in Section 4.3, Maat supports the exchange of verifiable reports. Because we require that detection is deterministic in Maat, as long as all messages are signed, these can be forwarded to any entity that supports Maat. Since the world model is not fundamentally ego-centric, it can also be used to perform detection at a central authority. A central authority can then determine the ground truth in cases where arbitration is required. Similarly, a misbehavior authority as in the US SCMS [168] can deploy Maat to determine which vehicle(s) to revoke based on received reports. Here it is important to note that the initial message processing phase in Maat can also link together pseudonyms if this information is available, by assigning all pseudonyms of the same vehicle to the same entity vertex.

Machine learning techniques Two challenges that many detectors face are the interpretation of locally collected data to find patterns (data-centric) and the merging of different misbehavior detectors into one coherent output (node-centric). To solve these problems, a number of schemes adapt ideas from machine learning. Similar ideas can also be observed in the domain of network intrusion detection, where patterns in traffic are identified through machine learning. The same can be seen in the domain of C-ITSs, where some authors are employing pattern recognition techniques and learned classifiers to determine message trustworthiness [71, 154]. In our view, the use of machine learning for this purpose should be followed critically, especially when the training phase of such classifiers includes explicit attacks that are classified. The difficulty with many machine learning approaches is two-fold: 1) training bias can result in invalid results when the training data does not properly cover the entire input space, and 2) the output of most algorithms is difficult to interpret. These are both issues that are being addressed separately in the field of machine learning; we point these out here because they are aggravated by the C-ITS context. The first, training bias, is aggravated in two ways by C-ITS: due to their experimental nature, there are no training sets available that describe all valid behavior in a C-ITS; additionally, the correctness of data is strongly influenced by context, a classifier for messages cannot be learned in isolation[1].

The second difficulty of machine learning is that algorithm output is difficult to interpret. Typical works only use the output as a classification between malicious and benign; in real-world scenarios, additional classes (e.g., malfunctioning components) might be very useful. More importantly, it is hard to judge based on the algorithm output why a certain message was classified as such. Due to the cyber-physical nature of C-ITSs, recovery & response decisions would be greatly simplified if such information was available (for example, "this vehicle claims false speeds"). Similarly, making revocation decisions and improving detection performance, which will be a necessary part of any real-world deployment of C-ITSs, require some human-interpretable output that can assist in figuring out *why* a detection was made.

As machine learning, subjective logic is also a technique borrowed from the field of artificial intelligence. As opposed to machine learning, however, subjective logic is more aimed at artificial reasoning, where a (human-interpretable) reasoning process is used. This significantly reduces one of the main the difficulties of machine learning for this application, namely that of output interpretation. However, artificial reasoning tools such as subjective logic also enable more understandable forms of machine learning. Maat provides new foundations for the application of machine learning in this domain; in our view, the best candidate is to learn optimal values of fusionability applied to detectors in different scenarios. Machine learning is useful here, because manually accounting for all possible scenarios in which different detector trust would be appropriate is infeasible. Because Maat also enables more extensive data collection and independently verifiable results (e.g., by a central authority), it presents a way in which learning can also be supplied with sufficient data. By performing learning tasks in the backend, human supervision can also be included to reduce the training biases mentioned above. However, a lot of practical work is still required for this to be useful.

[1] In short, this is important because the input space becomes much larger. For a simple classifier, the input space would only consist of all possible messages. However, in C-ITSs, the message validity depends on the previously received messages (e.g., vehicles cannot contain the same space at the same time, and the validity of a speed claim depends on the previous speed). Thus, the classifier must be stateful; if done naively, this creates an infinite input space (i.e., all message sequences). A classifier can thus only be learned on a small subset of the input space, and most classifiers applied in the literature do not address this issue explicitly.

Chapter 11

Future Work

This chapter provides an outlook of the thesis, discussing the relation between Maat and open issues in misbehavior detection for C-ITS and the application of Maat to other types of CPS. We also highlight some of the new research questions raided by Maat, which could be studied in future work.

11.1 Remaining Challenges for Maat

Maat, our framework for generic fusion of misbehavior detection mechanisms in cooperative intelligent transport systems, enables many novel features that were discussed in this thesis, but to some of these features, important limitations apply. We highlight these challenges in this section, discussing potential solutions that could be developed based on the results achieved in this thesis. Future work can address the resulting new and remaining challenges in future work.

11.1.1 Runtime Performance

In our work, we concentrated on detection performance metrics as a baseline for comparison with other schemes. This is in part because runtime performance cannot be assessed through the evaluation strategy we took, which is in line with the majority of work in this field, suggesting that this issue is considered insignificant at this stage. We designed Maat with the intent that it can perform well on modern multi-core systems by isolating detectors as much as possible from the framework itself. Our prototype executes in reasonable time any meaningful performance optimizations, and fast enough to perform large scale evaluations (around 37 million reception events) in a few hours with all parameterizations of Maat that we consider. However, we did not extensively study the detection latency or space requirements added by the framework as compared to executing the detectors independently. Existing hardware included in vehicles is rapidly evolving at this stage, as a variety of new applications is being deployed by vehicle manufacturers, and thus it is difficult to pin down what kind of hardware would be available to run Maat. We expect that as the available hardware improves, Maat will be combined with an increasing number of detectors, allowing it to scale up in terms of detection performance as new hardware becomes available. The validation functionality provided by Maat also enables expensive detection mechanisms to be executed offline by a central authority, e.g., for revocation purposes. Such advanced revocation features would allow individual vehicles to save computational resources by assuming attacks are less likely, under the assumption that sufficient evidence can still be gathered by the authority.

11.1.2 Sensor Data

Maat was originally developed to allow the inclusion of sensor data in the decision process. In our prototype, we focused on GPS information as the primary class of sensor data, because this data is very similar to the data received through ad-hoc communication from other vehicles. However, other sensor data may have significantly higher frequencies and the volume of data may be too high to store this data in the world model directly. Examples of such data include LIDAR scans and camera images. We did not consider these types of data directly, in particular because obtaining such sensor data that has a direct relationship with the data received through the C-ITS requires modeling and implementation

of sensors in the simulation environment. In addition, the volume of data is too large to include it into Maat directly. This, to include arbitrary sensor data into Maat, there are two complementary approaches; aggregated input or pruning within Maat. The first approach is to pre-aggregate sensor data by converting a series of camera or LIDAR images into a small set of points (such as labeled objects with properties such as position and speed), which can be processed individually by Maat. Secondly, as discussed briefly in Chapter 4, it is possible to extend Maat with a pruning component that regularly aggregates data items into a single data item. We consider these options to be complementary, together enabling a realistic extension to Maat. The second approach has the additional advantage of enabling Maat to contribute to secure aggregation and information dissemination. Finally, pruning within Maat enables the system to purge outdated, unnecessary data, which would be necessary to deploy Maat in the long term.

11.1.3 Expressiveness of Opinions

The current prototype of Maat provides a somewhat limited expressiveness compared to what subjective logic offers. In particular, we assume that any particular data item is either true or false, which is represented by a binomial subjective logic opinion. Although minor deviations from real-world information can be captured by a detector that produces opinions, some information is lost compared to the use of more complex opinion types. There is no theoretical boundary that prevents the implementation of Maat using multinomial or hyper opinions to express belief about more complex events. A more complex event is an event that has more than two possible states (e.g., the color of a vehicle). For subjective logic in general, it is necessary that the set of possible states is a finite set, although practical limitations mean that opinions on events with infinite possible states are not useful either way. However, for Maat, the difficulty additionally lies in the fact that multinomial opinions have $2k - 1$ degrees of freedom, and it is thus non-trivial to store these opinions efficiently. For hyper opinions, there are even more: these have $2^k + k - 3$ degrees of freedom. Limiting the domain of opinions (i.e., making k small) is a way to control the trade-off between expressiveness and storage requirements. Additional optimizations may be possible, e.g., by noting that in many domains, most of the beliefs will be zero, suggesting that optimized storage (e.g., only storing the degrees of freedom associated with states that have non-zero values) may be useful.

11.1.4 Expressiveness of Relationships

Maat currently assumes that each detector can evaluate the correctness of messages in isolation. This simplifying assumption enables us to provide a simple fusion process where all results are collected and fused as equal. However, some detectors do not actually determine the accuracy of a message, but rather relationships between messages. Detectors are often used as message classifiers, with the underlying and implicit assumption that messages received previously were correct. A simple example is the overlapping position detector designed originally by Bißmeyer et al. [45], which compares the positions of two messages and determines whether the vehicles overlap. This mechanism actually expresses that two messages cannot both be correct; the original authors resolve the conflict by picking the most trustworthy of the two senders. The Maat world model does not currently support the expression of opinions on multiple messages at once, but rather relies on the detector's state to express which of the two is more trustworthy.

We made an unsuccessful attempt to express these relationships through logical equations, which are a feature of subjective logic. However, it is non-trivial to find the correct logical equation that corresponds to each detector, because there are multiple representations for the same statement. In addition, adding data items as a combination of other data items increases Maat's storage complexity. In our implementation, we noticed that making decisions based on these equations is very susceptible to circular reasoning: if a decision depends on trusting one of two otherwise equally trustworthy messages, the order in which messages arrive matters. Similarly, making decisions based on equations potentially introduces an undesirable factor of dependence between detection mechanisms into Maat. This raises new questions and calls for the implementation of a reasoning component, whose design we consider to be future work. In the current version of Maat, we abstract away from equations by allowing detectors to generate and replace previous opinions with new versions. This concentrates the problem in the detector logic and removes information that could be useful in decision making.

11.2 New Research Questions

The challenges discussed above serve as ample inspiration for future developments in this area; in this section we provide new research questions relating to these challenges.

11.2.1 Subjective Logic: Opinion Representations & Fusion

Two of the primary limitations of Maat listed above are directly related to the practical application of subjective logic. Our attempts to apply subjective logic for a misbehavior detection system that autonomously decides on the validity of messages have revealed areas where subjective logic can be improved. The first of those areas was the design of multi-source fusion operations, which we provided in Chapter 5. However, as far as we are aware, there are no efficient implementations of multinomial or hyper opinions. Our implementation is based on existing source code from the original author of subjective logic, which only provides binomial opinions. We have made improvements to this code and made it publicly available[1]. As noted, the most important reason we identified for the lack of such an implementation is related to storage complexity. It thus appears intuitive to pose the question:

RQ 4. *Can a space-efficient implementation of multinomial opinions be designed, whose average complexity is significantly better than storing a Dirichlet PDF with* $2k + 1$ *parameters?*

A similar question exists for hyper opinions:

RQ 5. *Can a space-efficient implementation of hyper opinions be designed, whose average complexity is significantly better than storing a Dirichlet hyper-PDF with* $2^k + k - 1$ *parameters?*

Finally, we remark that our implementation of the consensus & compromise fusion (CCF) (Section 5.2.5) is computationally expensive for large input sets (i.e., large amount of detectors). This is caused by the fact that for n input opinions, there are many permutations of the domain of size k that satisfy the choices specified by our equations. Since the CCF is significantly more computationally expensive than the other fusion operations, application of this operation may not be worth the additional effort. However, an analysis of the relevant equations reveals on the one hand that CCF will likely perform better in multinomial and hyper opinion scenarios. Further analysis suggests that many of the conditional base rates defined there are by definition zero, and this leads us to believe that a much more computationally efficient implementation may be possible. It may thus also be reasonable to ask:

RQ 6. *What is the computational complexity of consensus & compromise fusion, and can the computational effort required be reduced?*

11.2.2 Subjective Logic: Data Relationships & Subjective Networks

Going beyond the representation and fusion of individual data items, we now address how subjective logic can express relationships in data. Recall that the domain of our subjective opinions is always represented by a single data item. However, many realistic detection mechanisms express opinions about relationships between data items, i.e., relations between different domains. Existing work typically circumvents this problem through the aforementioned implicit assumption that data already contained within the world model is true (as discussed in Chapter 7). For example, receiving a message from an attacker claiming to be at position P, followed by a legitimate message claiming to be at position P at approximately the same time causes the attacker message to be accepted and the legitimate message to be rejected. If the messages were reordered, the opposite would be true, at least for this specific detection mechanism. In Maat's current implementation it is assumed that each detection mechanism has an internal logic to consistently deal with this issue, however it raised the question whether this can be performed in a generic way:

RQ 7. *Is it possible to build a generic framework in which expressions on data conflicts can be stored independent of message order?*

[1] See `https://github.com/vs-uulm/subjective-logic-java`.

The immediate application of such a generic framework would be to be able to deal with the example situation above, and generically store that one of those two messages is correct. This information could then be fused with additional information about the correctness of these two messages (i.e., from other detectors) whenever an application accesses the data. An intuitive answer to this research question would be to extend Maat in such a way that it expresses data conflicts between any two data items. However, doing so greatly increases the storage complexity of Maat, and implies that the trustworthiness of every data item should be assessed to determine which is most accurate. Therefore, this also has a significant impact on the computational complexity, because it increases the graph size and adds additional comparisons of all fused opinions on the respective data items. Thus, another natural question becomes:

RQ 8. *How is the computational cost of fusion in Maat affected if relationships between data items are included in the world model?*

11.2.3 Real-world Experiments

A large body of work on vehicular communication and even cooperative intelligent transport systems has been performed using simulation experiments. Recent developments have shown wide-scale measurement campaigns for communication and applications, but as far as we are aware, none of those experiments cover security or misbehavior detection. A lot of research has thus been dedicated to perfecting simulation experiments, with significant strides being made by several of the major network and traffic simulation toolkits. These developments are essentially based on measurement campaigns and modeling work done in the various fields, as well as software implementations of the relevant communication layers (particularly MAC and multi-channel behavior). Combined with the evaluation methodology described in this thesis, we have discussed how datasets may be generated based on such simulation experiments to enable wide-scale studies of detection performance. Although such evaluations are very useful to assess detection performance in realistic settings, they are limited by the fact that simulations will not cover all real-world behavior. With recently performed large scale field trials, it may be time for the field of misbehavior detection to look towards real-world experiments for the C-ITS domain. Such real-world experiments will also potentially enable researchers to compare with other approaches to achieve reliability, such as sensor fusion. However, we note that real-world experiments for C-ITS are not necessarily sufficient to *prove* that misbehavior detection works, because of the nature of the attacks that are studied. Based on these ideas, we propose the following two research questions that could be investigated:

RQ 9. *How can realistic attacks be included in real-world field studies for the evaluation of misbehavior detection mechanisms?*

Note that real-world experiments with real-world attacks on applications can in the worst case lead to crashes, and thus we expect the primary means of study will be a modification or manipulation of existing test data. This can be done randomly, algorithmically or even through some learning process, depending on the desired attacker performance. Because field tests will likely not be as large as the possibilities for simulation results, we expect that it will also be important to consider how simulation can be improved through field trials. Therefore, we also pose this second question:

RQ 10. *How can misbehavior detection results from field trials of individual vehicles be generalized to analyze the effects of wide-scale deployment?*

11.3 Open Issues in Misbehavior Detection for C-ITS

Even though many specific aspects of misbehavior in C-ITSs have been addressed, are several notable open issues in this domain. These open issues were previously raised in our survey [170]; here we also address how Maat can potentially stimulate advances in each area.

Identification of misbehaving nodes Data-centric misbehavior detection mechanisms are essential for detecting conflicting data and identifying which data is malicious, but they may not be able to identify the attacker directly. Although some work, such as reliable platooning, the increased data reliability alone is enough, for many applications it is important to identify and remove the attackers

from the network in the long term. Identifying which node is behaving maliciously is useful for local or global revocation, limiting both the amount of attackers in the network and the impact they may have on the network. However, this requires low false positives: local exclusion of benign nodes could be catastrophic for both applications and future detection performance. In the worst case, it may lead to network segmentation, where trust mechanisms cause two groups of vehicles to mutually distrust each other. Therefore, it is important to both provide high quality detection and the ability to revise revocation decisions that were made. Maat supports such activities by storing the associated evidence. In addition, Maat could be combined with identification techniques (e.g., the Kalman filter tracking approach proposed by Stübing [108]) by associating multiple pseudonyms with the same node vertex. This process could also be performed after-the-fact by an authority that executes a central variant of Maat (see Section 4.3). The design privacy-preserving protocols that can efficiently detect who is behaving maliciously without de-anonymizing all affected users is an unsolved problem, and thus requires future work.

Voting, pseudonyms and Sybil attacks For local detection within a single vehicle, node-centric misbehavior detection mechanisms in the literature often rely on long-term identities to perform voting or other trust-based evaluation of data. The basic idea is simple: assuming that an event, such as an icy road, is detected by all passing vehicles and further assuming that the majority of vehicles is honest, a simple majority vote about whether the road is actually icy should reveal possible misbehavior. However, it is a common assumption that vehicles will use short-term pseudonyms for communication to protect privacy, as discussed in Section 2.1. Therefore, an important open issue is to build a scheme that enables voting with such pseudonyms, while still providing the necessary resilience against Sybil attacks, where an attacker uses multiple identities to inflate the trust in a vehicle or in some data. A simple solution would be to require that a vehicle can only use one pseudonym at a time, but this assumption requires significant cryptographic or communication overhead to protect the pseudonym keys. In addition, such an approach would greatly reduce the privacy that is achievable using pseudonyms. Another approach would be to focus on limited pseudonym re-use in combination, and essentially assume that detection mechanisms are good enough to detect attacks even in the presence of Sybil attacks. The implementation of such schemes in Maat could be an approach to study how feasible such an assumption is. The detected Sybil attacks should then, however, also incur strong penalties through a revocation infrastructure, since detecting Sybil attacks is known to be a hard problem.

Cooperative misbehavior detection As we have seen in our experiments in Chapter 8, local detection bias of individual vehicles is a significant issue. Some vehicles simply cannot detect certain attacks, if they are sufficiently well-crafted. Therefore, cooperative misbehavior detection may be the essential next step to detect even such challenging cases. Through local reporting, vehicles may be able to exchange evidence and detect more attacks than they otherwise could. However, we have to assume that the misbehavior reports can be fake as well. Some schemes propose to solve this issue by collecting a number of misbehavior reports from different vehicles, again using a voting mechanism, which relies on long-term identities, as discussed previously. Other approaches propose to solve the issue by so-called suicide schemes where vehicles reporting misbehavior of other vehicles also temporarily exclude themselves from the network (see Section 2.3.4). In our opinion, future work should focus on the integration of evidence received from other nodes, validating reports by re-running their misbehavior detection mechanisms. Comparing those results with internal detection results is very likely to reveal discrepancies caused by false reports; the main challenge for this approach is the required throughput to be able to transfer all evidence between all vehicles.

Level of reporting to back-end As discussed above, we need to rethink the level of reporting to the back-end in order to benefit from misbehavior detection in the back-end. If the back-end only receives reports about definite misbehavior, then local nodes have already identified the misbehavior. Hence, the additional benefit of involving the back-end is limited to possible revocation of the misbehaving nodes' certificates. On the other hand, it is not possible to report all data received by vehicles to the back-end because of bandwidth constraints. It is an open challenge to find a good trade-off between reporting somewhat suspicious behavior to the back-end in order to allow for better attack detection and not using too much bandwidth.

11.4 Applications beyond C-ITS: Misbehavior Detection in ICS

Now that we have studied the state of the art misbehavior detection mechanisms for C-ITSs and designed a framework for their fusion, we address the possible application of these ideas to other cyber-physical system (CPS). For a detailed discussion of different types of CPS and the security work done within those fields, refer to Giraldo et al. [155]. In this section, we take the example of industrial control system (ICS), and discuss the experiments we performed in that context with an earlier version of Maat, as well as the lessons learned in this process.

11.4.1 Industrial Control Systems

Unlike C-ITSs, ICSs are a type of CPSs that has historically grown out of the field of control systems, and the focus of these systems has primarily been safety, rather than security [119]. Specifically, ICSs must avoid individual but random failures from escalating and causing a *cascading failure*, that is, causing a shutdown of components that are not directly related to the original failure. To this end, much research has been conducted, but recent developments regarding attacks on these systems, such as those by Stuxnet, that these systems are not built to resist targeted attacks. In the past, ICSs were internal networks, which were completely separated from the Internet by a so-called "air-gap", meaning no direct connectivity is possible. However, the necessity of patching these systems and the use of re-writable media like USB-devices has led many security researchers to conclude that a complete "air-gap" is not a feasible solution. In addition, secure connectivity has already provided organizations with significant cost savings and ease of management for these systems. Therefore, we study how misbehavior detection can be used to improve security in these systems, in particular against an insider attack that attempts to cause damage by disrupting or sabotaging the industrial process controlled by the ICS. Again, the recurring patterns we have identified for C-ITSs may provide significant insights for new developments in ICS, for example through the application of physical models to detect anomalies with high accuracy. The open issues faced in C-ITSs similarly affect ICSs; a lack of public, real-world datasets that are detailed enough to detect misbehavior specifically is very important additional challenge, which we also observed in C-ITSs. Combined with the fact that most detection mechanisms are closed-source, this makes reproducibility and comparison between mechanisms and design of suitable misbehavior detection fusion nearly impossible. Worse, ICSs additionally face the challenge that there is virtually no authentication of any kind available in most networks, which means that misbehavior detection must also detect inauthentic messages. In our view, misbehavior detection mechanisms are not the suitable tool to solve such authentication problems; proper authentication and integrity is necessary to build a reasonable system.

11.4.2 Application of Maat to ICS

As part of this thesis, we experimented with the application of an early prototype of Maat to an ICSs testbed as part of the CRISALIS research project. The result of this analysis was that misbehavior detection in such a testbed requires that there is more than once source of information that is fed to Maat. Unlike the scenario of C-ITSs, where vehicle behavior is well-defined and can be converted into data-centric detection mechanisms easily, in an ICS such attempts at modeling the system essentially end up replicating the control logic. We therefore attempted to include a secondary source of information in the form of a host-based intrusion detection system (HIDS) that sends alerts to Maat over a secure channel.

The testbed we used was provided by a project partner, and simulated a power grid as an industrial control system. The testbed includes typical components of an ICS, consisting of two programmable logic controllers (PLCs), a human machine interface (HMI), a switch and a SCADA workstation. A PLC is a hardware component in which controllers can be implemented that describe an industrial process; we discuss the power grid control in some detail below. There is also a HMI, which is a simple hardware device typically installed in a factory or power system, which provides a simple display for an operator that supervises the controlled process. The SCADA workstation is a Windows machine running the proprietary tools needed to program and flash the other devices; it can also be used as a monitoring tool. The testbed itself models a power grid that receives input from solar and wind sources, as well as power consumption from houses. These three inputs are used by the distributed controller to decide the output of a power plant; the original demonstration of this testbed shows how attacks on these PLCs can be used

to attack the power plant to cause over production. For communication between the distributed parts of the PLCs, a simple switch is used to connect all the components over a standard TCP/IP network. Although the documentation claims the Modbus protocol can be used, all devices in the testbed we had access to used the proprietary Siemens S7 protocol to communicate over TCP. The S7 protocol is widely deployed, but also known to be vulnerable to a variety of attacks and protocol issues [63] (e.g., lacking authentication).

In our application of Maat to the testbed, we designed a world model for this control process. The model of this process consists of a set of data items, each of which has an associated source, and these data items each represent a specific variable in the control process. A set of detectors then evaluates the data items stored in the framework. The detectors are are PLC command validation, the raw S7 packet plausibility check, and an alert report from a HIDS that reports to Maat when it detects suspicious behavior on the Windows machine that supervises the controller. The PLC command validation checks whether an observed PLC command (i.e., increase or decrease output) corresponds to the desired output based on previously received messages. The raw S7 packets are checked for value plausibility (a simple range check). The HIDS uses a secure channel to alert Maat of suspicious behavior on the SCADA machine.

During the on-site analysis with the testbed, we attempted to perform experiments to show that Maat could be applied in an ICS. Unfortunately, technical difficulties with the testbed resulted in a control behavior that was entirely specified by the controlling PLC, which performed direct measurements. This is different from what was documented for the testbed; the original design foresaw, as discussed above, that the controller receives its inputs through the network. On-site analysis over several days established that none of the components in the network were able to successfully establish TCP connections, but the process control behaved normally. Because the original developers of the control process were no longer available, and the proprietary S7 protocol could not be replaced, we decided to abandon the use of this testbed.

Lessons Learned First and most importantly, we learned that performing misbehavior detection in the absence of authenticity is very difficult, especially when the detection system is deployed to the same switch as the controller. When deployed in this manner, Maat only has a single source of information (i.e., the network), whose authenticity is not guaranteed. This means that Maat must rely entirely on data-centric mechanisms; however, if integrity is also not guaranteed, a data-centric analysis of sensor data exchanged through the network is not particularly helpful, other than to decide whether this data can at some point occur in the process. On the other hand, in C-ITSs, instances of Maat always have access to at least some information sources that are not attacker-controlled (e.g., GPS data or signal measurements). Another lesson learned is that detectors in ICS will often very closely resemble the processes they validate, since the process is typically controlled in a centralized manner. This is fundamentally distinct from C-ITSs and even fully autonomous driving: these systems are fundamentally decentralized, since locality is important. The fact that detectors closely resemble the controlled physical process is seen as problematic in this field, where some suggest simple redundancy could provide similar benefits at much lower cost [119]. Finally, our idea to integrate a HIDS into Maat suggested that its' fusion algorithms may also be useful for Security Information and Event Management (SIEM) systems. These systems are used in large enterprise networks to monitor and track attackers under human supervision, but typically focus on high-level information fusion, if they fuse data at all. Maat's use of low-level fusion techniques may be beneficial for such systems if they are used to monitor CPSs, which could be studied in future work.

11.5 Summary

This thesis has provided new insights on misbehavior detection in the context of C-ITSs, in particular with regards to result fusion. Using subjective logic and a flexible graph-based data analysis framework called Maat, we describe a complete fusion framework that can be used as a stepping stone for misbehavior response. This response can also benefit from the uncertainty information that Maat provides, as well as the reporting mechanisms that Maat enables. Beyond future applications, we have also introduced new research questions raised by our work, highlighting how Maat's foundations may be extended to enable advanced detection techniques.

Bibliography

[1] AP Dempster. "Upper and lower probabilities induced by a multivalued mapping". In: *The annals of mathematical statistics* 38.2 (1967), pages 325–339. DOI: 10.1214/aoms/1177698950.

[2] G Shafer. *A mathematical theory of evidence*. Princeton university press, 1976. ISBN: 978-0691100425.

[3] D Dolev and ACC Yao. "On the security of public key protocols". In: *IEEE Transactions on Information Theory* 29.2 (March 1983), pages 198–208. DOI: 10.1109/TIT.1983.1056650.

[4] LA Zadeh. "Review of a mathematical theory of evidence". In: *AI magazine* 5.3 (1984), page 81. DOI: 10.1609/aimag.v5i3.452.

[5] HC Joksch. "Velocity change and fatality risk in a crash – a rule of thumb". In: *Accident Analysis & Prevention* 25.1 (1993), pages 103–104.

[6] A Jøsang. "Artificial Reasoning with Subjective Logic". In: *Proceedings of the Second Australian Workshop on Commonsense Reasoning*. 1997. URL: http://folk.uio.no/josang/papers/Jos1997-AWCR.pdf.

[7] H Debar, M Dacier, and A Wespi. "Towards a taxonomy of intrusion-detection systems". In: *Computer Networks* 31.8 (1999), pages 805–822. DOI: 10.1016/S1389-1286(98)00017-6.

[8] S Marti, TJ Giuli, K Lai, and M Baker. "Mitigating routing misbehavior in mobile ad hoc networks". In: *Proceedings of the 6th annual international conference on Mobile computing and networking (MobiCom)*. MobiCom '00. Boston, Massachusetts, USA: ACM Press, 2000, pages 255–265. DOI: 10.1145/345910.345955.

[9] JR Douceur. "The Sybil Attack". en. In: *Peer-to-Peer Systems*. Edited by P Druschel, F Kaashoek, and A Rowstron. Volume 2429. Lecture Notes in Computer Science. Springer Berlin Heidelberg, March 2002, pages 251–260. DOI: 10.1007/3-540-45748-8_24.

[10] DE Gaylor and EG Lightsey. "GPS/INS Kalman Filter Design for Spacecraft Operating in the Proximity of the International Space Station". In: *AIAA Guidance, Navigation, and Control Conference and Exhibit, Guidance, Navigation, and Control and Co-located Conferences*. 2003. DOI: 10.2514/6.2003-5445.

[11] JM Estévez-Tapiador, P Garcia-Teodoro, and JE Díaz-Verdejo. "Anomaly detection methods in wired networks: a survey and taxonomy". In: *Computer Communications* 27.16 (2004), pages 1569–1584. DOI: 10.1016/j.comcom.2004.07.002.

[12] P Golle, D Greene, and J Staddon. "Detecting and correcting malicious data in VANETs". In: *Proceedings of the first ACM workshop on Vehicular ad hoc networks (VANET)*. New York, NY, USA: ACM Press, 2004, page 29. DOI: 10.1145/1023875.1023881.

[13] T Leinmüller, A Held, G Schäfer, and A Wolisz. *Intrusion Detection in VANETs*. Proceedings of 12th IEEE International Conference on Network Protocols (ICNP 2004) – Student Poster Session. October 2004. URL: http://www.leinmueller.de.

[14] A Lazarevic, V Kumar, and J Srivastava. "Intrusion Detection: A Survey". In: *Managing Cyber Threats*. Edited by V Kumar, J Srivastava, and A Lazarevic. Volume 5. Massive Computing. Springer US, 2005, pages 19–78. DOI: 10.1007/0-387-24230-9_2.

[15] AA Cárdenas, JS Baras, and K Seamon. "A framework for the evaluation of intrusion detection systems". In: *IEEE Symposium on Security and Privacy*. IEEE, 2006. DOI: 10.1109/SP.2006.2.

[16] J Davis and M Goadrich. "The Relationship Between Precision-Recall and ROC Curves". In: *Proceedings of the 23rd International Conference on Machine Learning*. ICML '06. Pittsburgh, Pennsylvania, USA: ACM, 2006, pages 233–240. DOI: 10.1145/1143844.1143874.

[17] T Leinmüller, C Maihöfer, E Schoch, and F Kargl. "Improved security in geographic ad hoc routing through autonomous position verification". In: *Proceedings of the 3rd international workshop on Vehicular ad hoc networks (VANET)*. New York, NY, USA: ACM Press, 2006, pages 57–66. DOI: 10.1145/1161064.1161075.

[18] M Raya, P Papadimitratos, and JP Hubaux. "Securing Vehicular Communications". In: *Wireless Communications Magazine* 13.5 (October 2006), pages 8–15. DOI: 10.1109/WC-M.2006.250352.

[19] "Securing Vehicular Communications – Assumptions, Requirements, and Principles". In: *Workshop on Embedded Security in Cars (ESCAR)*. November 2006.

[20] B Xiao, B Yu, and C Gao. "Detection and Localization of Sybil Nodes in VANETs". In: *Proceedings of the 2006 workshop on Dependability issues in wireless ad hoc networks and sensor networks (DIWANS)*. New York, NY, USA: ACM Press, 2006, pages 1–8. DOI: 10.1145/1160972.1160974.

[21] G Guette and B Ducourthial. "On the Sybil Attack Detection in VANET". In: *2007 IEEE Internatonal Conference on Mobile Adhoc and Sensor Systems*. IEEE, October 2007, pages 1–6. DOI: 10.1109/MOBHOC.2007.4428742.

[22] T Leinmüller, E Schoch, and C Maihöfer. "Security requirements and solution concepts in vehicular ad hoc networks". In: *Fourth Annual Conference on Wireless on Demand Network Systems and Services (WONS)*. IEEE, January 2007, pages 84–91. DOI: 10.1109/WONS.2007.340489.

[23] NW Lo and HC Tsai. "Illusion Attack on VANET Applications - A Message Plausibility Problem". In: *2007 IEEE Globecom Workshops*. IEEE, November 2007, pages 1–8. DOI: 10.1109/GLOCOMW.2007.4437823.

[24] M Raya and JP Hubaux. "Securing vehicular ad hoc networks". In: *Journal of Computer Security* 15 (2007), pages 39–68. DOI: 10.3233/JCS-2007-15103.

[25] M Raya, P Papadimitratos, I Aad, D Jungels, and JP Hubaux. "Eviction of Misbehaving and Faulty Nodes in Vehicular Networks". In: *Journal on Selected Areas in Communications* 25.8 (October 2007), pages 1557–1568. DOI: 10.1109/JSAC.2007.071006.

[26] N Banerjee, MD Corner, D Towsley, and BN Levine. "Relays, base stations, and meshes: enhancing mobile networks with infrastructure". In: *Proceedings of the 14th ACM international conference on Mobile computing and networking*. MobiCom '08. San Francisco, California, USA: ACM Press, 2008, pages 81–91. DOI: 10.1145/1409944.1409955.

[27] Z Cao, J Kong, U Lee, M Gerla, and Z Chen. "Proof-of-Relevance : Filtering False Data via Authentic Consensus in Vehicle Ad-hoc Networks". In: *IEEE INFOCOM Workshops*. IEEE, 2008, pages 1–6. DOI: 10.1109/INFOCOM.2008.4544650.

[28] F Kargl, P Papadimitratos, L Buttyan, M Müter, E Schoch, B Wiedersheim, TV Thong, G Calandriello, A Held, A Kung, and JP Hubaux. "Secure vehicular communication systems: implementation, performance, and research challenges". In: *Communications Magazine* 46.11 (2008), pages 110–118. DOI: 10.1109/MCOM.2008.4689253.

[29] T Leinmüller, RK Schmidt, E Schoch, A Held, and G Schäfer. "Modeling Roadside Attacker Behavior in VANETs". In: *2008 IEEE GLOBECOM Workshops*. IEEE, 2008, pages 1–10. DOI: 10.1109/GLOCOMW.2008.ECP.63.

[30] T Leinmüller, E Schoch, F Kargl, and C Maihöfer. "Decentralized position verification in geographic ad hoc routing". In: *Security and Communication Networks* 3.4 (2008), pages 289–302. DOI: 10.1002/sec.56.

[31] X Lin, R Lu, C Zhang, H Zhu, PH Ho, and X Shen. "Security in vehicular ad hoc networks". In: *Communications Magazine* 46.4 (2008), pages 88–95. DOI: 10.1109/MCOM.2008.4481346.

[32] P Papadimitratos, L Buttyan, T Holczer, E Schoch, J Freudiger, M Raya, Z Ma, F Kargl, A Kung, and JP Hubaux. "Secure vehicular communication systems: design and architecture". In: *Communications Magazine* 46.11 (2008), pages 100–109.

[33] M Raya, P Papadimitratos, VD Gligor, and JP Hubaux. "On Data-Centric Trust Establishment in Ephemeral Ad Hoc Networks". In: *2008 IEEE INFOCOM - The 27th Conference on Computer Communications*. IEEE, April 2008, pages 1238–1246. DOI: 10.1109/INFOCOM.2008.180.

[34] F Sabahi and A Movaghar. "Intrusion Detection: A Survey". In: *3rd International Conference on Systems and Networks Communications (ICSNC)*. IEEE, 2008, pages 23–26. DOI: 10.1109/ICSNC.2008.44.

[35] RK Schmidt, T Leinmüller, E Schoch, A Held, and G Schäfer. "Vehicle Behavior Analysis to Enhance Security in VANETs". In: *Proceedings of the 4th Workshop on Vehicle to Vehicle Communications (V2VCOM 2008)*. IEEE, 2008, pages 1–8. URL: http://www.leinmueller.de/lib/exe/fetch.php/publications/slshs08vebas.pdf.

[36] E Schoch, F Kargl, and M Weber. "Communication patterns in VANETs". In: *Communications Magazine* 46.11 (2008), pages 119–125. DOI: 10.1109/MCOM.2008.4689254.

[37] G Yan, S Olariu, and MC Weigle. "Providing VANET Security Through Active Position Detection". In: *Computer Communications* 31.12 (July 2008), pages 2883–2897. DOI: 10.1016/j.comcom.2008.01.009.

[38] C Chen, X Wang, W Han, and B Zang. "A Robust Detection of the Sybil Attack in Urban VANETs". In: *29th IEEE International Conference on Distributed Computing Systems Workshops*. IEEE, June 2009, pages 270–276. DOI: 10.1109/ICDCSW.2009.48.

[39] A Hamieh, J Ben-Othman, and L Mokdad. "Detection of Radio Interference Attacks in VANET". In: *IEEE Global Telecommunications Conference (GLOBECOM)*. IEEE, 2009, pages 1–5. DOI: 10.1109/GLOCOM.2009.5425381.

[40] M Raya. "Data-centric trust in ephemeral networks". PhD thesis. Lausanne: EPFL, 2009. DOI: 10.5075/epfl-thesis-4423.

[41] SAE International. *Dedicated Short Range Communications (DSRC) Message Set Dictionary*. Technical report J2735. November 2009. URL: http://standards.sae.org/j2735_200911/.

[42] TL Willke, P Tientrakool, and NF Maxemchuk. "A survey of inter-vehicle communication protocols and their applications". In: *IEEE Communications Surveys & Tutorials* 11.2 (2009), pages 3–20. DOI: 10.1109/SURV.2009.090202.

[43] X Zhuo, J Hao, D Liu, and Y Dai. "Removal of misbehaving insiders in anonymous VANETs". In: *Proceedings of the 12th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems - MSWiM '09*. New York, NY, USA: ACM Press, 2009, page 106. DOI: 10.1145/1641804.1641824.

[44] I Bilogrevic, MH Manshaei, M Raya, and JP Hubaux. "Optimal revocations in ephemeral networks: A game-theoretic framework". In: *Proceedings of the 8th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*. IEEE, June 2010, pages 21–30.

[45] N Bißmeyer, C Stresing, and KM Bayarou. "Intrusion Detection in VANETs Through Verification of Vehicle Movement Data". In: *Proceedings of the Vehicular Networking Conference (VNC)*. IEEE, 2010, pages 166–173. DOI: 10.1109/VNC.2010.5698232.

[46] S Dietzel, E Schoch, B Könings, M Weber, and F Kargl. "Resilient secure aggregation for vehicular networks". In: *IEEE Network* 24.1 (January 2010), pages 26–31. DOI: 10.1109/MNET.2010.5395780.

[47] ETSI. *Intelligent Transport Systems (ITS); Communications Architecture*. http://www.etsi.org/deliver/etsi_en/302600_302699/302665/01.01.01_60/en_302665v010101p.pdf. Last Accessed 2018-04-19. September 2010.

[48] ETSI. *Intelligent Transport Systems (ITS); Vehicular communications; GeoNetworking; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service*. Technical Specification: TS 102 637-3, V1.1.1. September 2010.

[49] M Ghosh, A Varghese, A Gupta, AA Kherani, and SN Muthaiah. "Detecting misbehaviors in VANET with integrated root-cause analysis". In: *Ad Hoc Networks* 8.7 (September 2010), pages 778–790. DOI: 10.1016/j.adhoc.2010.02.008.

[50] RKS Hankin. "A Generalization of the Dirichlet Distribution". In: *Journal of Statistical Software* 33 (11 2010). DOI: 10.18637/jss.v033.i11.

[51] J Hortelano, JC Ruiz, and P Manzoni. "Evaluating the Usefulness of Watchdogs for Intrusion Detection in VANETs". In: *IEEE International Conference on Communications Workshops (ICC)*. IEEE, May 2010, pages 1–5. DOI: 10.1109/ICCW.2010.5503946.

[52] A Jøsang, J Diaz, and M Rifqi. "Cumulative and averaging fusion of beliefs". In: *Information Fusion* 11.2 (April 2010), pages 192–200. DOI: 10.1016/j.inffus.2009.05.005.

[53] THJ Kim, A Studer, R Dubey, X Zhang, A Perrig, F Bai, B Bellur, and A Iyer. "VANET alert endorsement using multi-source filters". In: *Proceedings of the seventh ACM international workshop on VehiculAr InterNETworking (VANET)*. New York, NY, USA: ACM Press, 2010, page 51. DOI: 10.1145/1860058.1860067.

[54] K Koscher, A Czeskis, F Roesner, S Patel, T Kohno, S Checkoway, D McCoy, B Kantor, D Anderson, H Shacham, and S Savage. "Experimental Security Analysis of a Modern Automobile". In: *2010 IEEE Symposium on Security and Privacy*. May 2010, pages 447–462. DOI: 10.1109/SP.2010.34.

[55] M Larburu, J Sanchez, and DJ Rodriguez. "Safe Road Trains for Environment: Human factors' aspects in dual mode transport systems". In: *Proceedings of 17th ITS World Congress*. 2010. URL: https://trid.trb.org/view/1136597.

[56] T Leinmüller, RK Schmidt, and A Held. "Cooperative Position Verification-Defending Against Roadside Attackers 2.0". In: *Proceedings of 17th ITS World Congress*. 2010, pages 1–8. URL: https://trid.trb.org/view/1128488.

[57] B Liu, JT Chiang, and Yc Hu. *Limits on Revocation in VANETs*. Pre-proceedings of the 8th International Conference on Applied Cryptography and Network Security (ACNS 2010). 2010. URL: http://users.crhc.illinois.edu/yihchun/pubs/acns10.pdf.

[58] M Raya, R Shokri, and JP Hubaux. "On the tradeoff between trust and privacy in wireless ad hoc networks". In: *Proceedings of the third ACM conference on Wireless network security*. WiSec '10. Hoboken, New Jersey, USA: ACM Press, 2010, pages 75–80. DOI: 10.1145/1741866.1741879.

[59] F Schaub, F Kargl, Z Ma, and M Weber. "V-Tokens for Conditional Pseudonymity in VANETs". In: *Proceedings of the Wireless Communications and Networking Conference (WCNC)*. IEEE, April 2010, pages 1–6. DOI: 10.1109/WCNC.2010.5506126.

[60] R Sommer and V Paxson. "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection". In: *2010 IEEE Symposium on Security and Privacy*. May 2010, pages 305–316. DOI: 10.1109/SP.2010.25.

[61] H Stübing, A Jaeger, N Bißmeyer, C Schmidt, and SA Huss. "Verifying mobility data under privacy considerations in Car-to-X communication". In: *Proceedings of 17th ITS World Congress*. ITS America, 2010, pages 1–12. URL: http://trid.trb.org/view.aspx?id=1107457.

[62] B Wiedersheim, Z Ma, F Kargl, and P Papadimitratos. "Privacy in inter-vehicular networks: why simple pseudonym change is not enough". In: *Proceedings of the 7th international conference on Wireless on-demand network systems and services*. WONS'10. Kranjska Gora, Slovenia: IEEE Press, 2010, pages 176–183. ISBN: 978-1-4244-6059-5. URL: http://dl.acm.org/citation.cfm?id=1834182.1834216.

[63] D Beresford. "Exploiting Siemens Simatic S7 PLCs". In: *Blackhat USA* (2011). URL: https://media.blackhat.com/bh-us-11/Beresford/BH_US11_Beresford_S7_PLCs_WP.pdf.

[64] I Bilogrevic, MH Manshaei, M Raya, and JP Hubaux. "OREN: Optimal revocations in ephemeral networks". In: *Computer Networks* 55.5 (2011), pages 1168–1180. DOI: 10.1016/j.comnet.2010.11.010.

[65] N Bißmeyer, B Schünemann, I Radusch, and C Schmidt. "Simulation of attacks and corresponding driver behavior in vehicular ad hoc networks with VSimRTI". In: *Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques*. SIMUTools '11. ICST, Brussels, Belgium, Belgium: ICST, 2011, pages 162–167. ISBN: 978-1-936968-00-8. URL: http://dl.acm.org/citation.cfm?id=2151054.2151086.

[66] G Calandriello, P Papadimitratos, JP Hubaux, and A Lioy. "On the Performance of Secure Vehicular Communication Systems". In: *IEEE Transactions on Dependable and Secure Computing (IEEE TDSC)* 8.6 (November 2011), pages 898–912. DOI: 10.1109/TDSC.2010.58.

[67] ETSI. *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Local Dynamic Map (LDM); Rationale for and guidance on standardization.* Technical report TR 102 863, V1.1.1. ETSI, 2011.

[68] ETSI. *Intelligent Transport Systems (ITS); Vehicular communications; GeoNetworking; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service.* Technical Specification: TS 102 637-2, V1.1.1. March 2011.

[69] ETSI. *Intelligent Transport Systems (ITS); Vehicular communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 1: Media-Independent Functionality.* Technical Specification: TS 102 636-4-1, V1.1.1. June 2011.

[70] J Grover, MS Gaur, V Laxmi, and NK Prajapati. "A Sybil Attack Detection Approach using Neighboring Vehicles in VANET". In: *Proceedings of the 4th international conference on Security of information and networks (SIN)*. New York, NY, USA: ACM Press, 2011, page 151. DOI: 10.1145/2070425.2070450.

[71] J Grover, NK Prajapati, V Laxmi, and MS Gaur. "Machine Learning Approach for Multiple Misbehavior Detection in VANET". In: *Advances in Computing and Communications*. Edited by A Abraham, JL Mauri, JF Buford, J Suzuki, and SM Thampi. Volume 192. Communications in Computer and Information Science. Springer Berlin Heidelberg, 2011, pages 644–653. DOI: 10.1007/978-3-642-22720-2_68.

[72] HC Hsiao, A Studer, R Dubey, E Shi, and A Perrig. "Efficient and secure threshold-based event validation for VANETs". In: *Proceedings of the fourth ACM conference on Wireless network security*. WiSec '11. Hamburg, Germany: ACM Press, 2011, pages 163–174. DOI: 10.1145/1998412.1998440.

[73] A Jaeger, N Bißmeyer, H Stübing, and SA Huss. "A Novel Framework for Efficient Mobility Data Verification in Vehicular Ad-hoc Networks". In: *International Journal of Intelligent Transportation Systems Research* 10.1 (August 2011), pages 11–21. DOI: 10.1007/s13177-011-0038-9.

[74] P Kleberger, T Olovsson, and E Jonsson. "Security aspects of the in-vehicle network in the connected car". In: *2011 IEEE Intelligent Vehicles Symposium (IV)*. June 2011, pages 528–533. DOI: 10.1109/IVS.2011.5940525.

[75] J Petit, M Feiri, and F Kargl. "Spoofed data detection in VANETs using dynamic thresholds". In: *Vehicular Networking Conference (VNC)*. IEEE, November 2011, pages 25–32. DOI: 10.1109/VNC.2011.6117120.

[76] DB Rawat, BB Bista, G Yan, and MC Weigle. "Securing Vehicular Ad-hoc Networks Against Malicious Drivers: A Probabilistic Approach". In: *Proceedings of the 2011 International Conference on Complex, Intelligent, and Software Intensive Systems*. IEEE, June 2011, pages 146–151. DOI: 10.1109/CISIS.2011.30.

[77] J Rezgui and S Cherkaoui. "Detecting faulty and malicious vehicles using rule-based communications data mining". In: *Proceedings of the 36th Conference on Local Computer Networks (LCN)*. IEEE, October 2011, pages 827–834. DOI: 10.1109/LCN.2011.6115558.

[78] DA Rivas, JM Barceló-Ordinas, MG Zapata, and JD Morillo-Pozo. "Security on VANETs: Privacy, misbehaving nodes, false information and secure data aggregation". In: *Journal of Network and Computer Applications* 34.6 (November 2011), pages 1942–1955. DOI: 10.1016/j.jnca.2011.07.006.

[79] S Ruj, MA Cavenaghi, Z Huang, A Nayak, and I Stojmenovic. "On Data-Centric Misbehavior Detection in VANETs". In: *Proceedings of the IEEE Vehicular Technology Conference (VTC Fall)*. IEEE, September 2011, pages 1–5. DOI: 10.1109/VETECF.2011.6093096.

[80] C Sommer, R German, and F Dressler. "Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis". In: *IEEE Transactions on Mobile Computing* 10.1 (January 2011), pages 3–15. DOI: 10.1109/TMC.2010.133.

[81] R Stanica, E Chaput, and AL Beylot. "Simulation of Vehicular Ad-hoc Networks: Challenges, Review of Tools and Recommendations". In: *Computer Networks* 55.14 (2011), pages 3179–3188. DOI: http://dx.doi.org/10.1016/j.comnet.2011.05.007.

[82] H Stübing, J Firl, and SA Huss. "A two-stage verification process for Car-to-X mobility data based on path prediction and probabilistic maneuver recognition". In: *2011 IEEE Vehicular Networking Conference (VNC)*. IEEE, November 2011, pages 17–24. DOI: 10.1109/VNC.2011.6117119.

[83] C Troncoso, G Danezis, E Kosta, J Balasch, and B Preneel. "PriPAYD: Privacy-Friendly Pay-As-You-Drive Insurance". In: *Transactions on Dependable and Secure Computing* 8.5 (2011), pages 742–755. DOI: 10.1109/TDSC.2010.71.

[84] P Victor, M De Cock, and C Cornelis. "Trust and Recommendations". In: *Recommender Systems Handbook*. Edited by F Ricci, L Rokach, B Shapira, and PB Kantor. Boston, MA: Springer US, 2011, pages 645–675. DOI: 10.1007/978-0-387-85820-3_20.

[85] G Yan, BB Bista, DB Rawat, and EF Shaner. "General Active Position Detectors Protect VANET Security". In: *Proceedings of the 2011 International Conference on Broadband and Wireless Computing, Communication and Applications*. IEEE, October 2011, pages 11–17. DOI: 10.1109/BWCCA.2011.12.

[86] T Zhou, RR Choudhury, P Ning, and K Chakrabarty. "P2DAP — Sybil Attacks Detection in Vehicular Ad Hoc Networks". In: *Journal on Selected Areas in Communications* 29.3 (March 2011), pages 582–594. DOI: 10.1109/JSAC.2011.110308.

[87] RP Barnwal and SK Ghosh. "Heartbeat message based misbehavior detection scheme for vehicular ad-hoc networks". In: *International Conference on Connected Vehicles and Expo (ICCVE)*. IEEE. 2012, pages 29–34. DOI: 10.1109/ICCVE.2012.14.

[88] N Bißmeyer, S Mauthofer, KM Bayarou, and F Kargl. "Assessment of node trustworthiness in VANETs using data plausibility checks with particle filters". In: *Vehicular Networking Conference (VNC), 2012 IEEE*. IEEE. 2012, pages 78–85. DOI: 10.1109/VNC.2012.6407448.

[89] N Bißmeyer, J Njeukam, J Petit, and KM Bayarou. "Central misbehavior evaluation for VANETs based on mobility data plausibility". In: *Proceedings of the ninth ACM international workshop on Vehicular inter-networking, systems, and applications (VANET)*. New York, NY, USA: ACM Press, 2012, pages 73–82. DOI: 10.1145/2307888.2307902.

[90] S Chang, Y Qi, H Zhu, J Zhao, and X Shen. "Footprint: Detecting Sybil Attacks in Urban Vehicular Networks". In: *Transactions on Parallel and Distributed Systems* 23.6 (2012), pages 1103–1114. DOI: 10.1109/TPDS.2011.263.

[91] 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. Low Wood Bay, Lake District, UK: ACM Press, 2012, pages 135–138. DOI: 10.1145/2307888.2307915.

[92] M McGurrin. *Vehicle Information Exchange Needs for Mobility Applications Exchange*. Technical report. United States Department of Transportation, February 2012. URL: http://ntl.bts.gov/lib/45000/45400/45492/Vehicle_Information_Needs_and_BSM_-_FINAL_508Compliant.pdf.

[93] O Puñal, A Aguiar, and J Gross. "In VANETs We Trust?: Characterizing RF Jamming in Vehicular Networks". In: *Proceedings of the Ninth ACM International Workshop on Vehicular Inter-networking, Systems, and Applications*. VANET '12. New York, NY, USA: ACM, 2012, pages 83–92. DOI: 10.1145/2307888.2307903.

[94] R Rajamani. *Vehicle Dynamics and Control*. Springer US, 2012. DOI: 10.1007/978-1-4614-1433-9.

[95] A Tchamova and J Dezert. "On the behavior of Dempster's rule of combination and the foundations of Dempster-Shafer Theory". In: *2012 6th IEEE International Conference Intelligent Systems*. September 2012, pages 108–113. DOI: 10.1109/IS.2012.6335122.

[96] S Amin, GA Schwartz, and A Hussain. "In Quest of Benchmarking Security Risks to Cyber-Physical Systems". In: *IEEE Network* 27.1 (January 2013), pages 19–24. DOI: 10.1109/MNET.2013.6423187.

[97] N Bißmeyer, J Petit, and KM Bayarou. "Copra: Conditional pseudonym resolution algorithm in VANETs". In: *2013 10th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*. March 2013, pages 9–16. DOI: 10.1109/WONS.2013.6578314.

[98] N Bißmeyer, KH Schröder, J Petit, S Mauthofer, and KM Bayarou. "Short paper: Experimental analysis of misbehavior detection and prevention in VANETs". In: *Vehicular Networking Conference*. IEEE, 2013, pages 198–201. DOI: 10.1109/VNC.2013.6737612.

[99] ETSI. *ETSI TS 103 097: Intelligent Transport Systems (ITS); Security; Security header and certificate formats*. Technical Specification: TS 103 097, V1.1.1. April 2013.

[100] M Fiore, CE Casetti, CF Chiasserini, and P Papadimitratos. "Discovery and Verification of Neighbor Positions in Mobile Ad Hoc Networks". In: *IEEE Transactions on Mobile Computing (IEEE TMC)* 12.2 (February 2013), pages 289–303. DOI: 10.1109/TMC.2011.258.

[101] RM Gerdes, C Winstead, and K Heaslip. "CPS: An Efficiency-motivated Attack Against Autonomous Vehicular Transportation". In: *Proc. Annual Computer Security Applications Conference*. ACSAC '13. New Orleans, Louisiana, USA: ACM, 2013, pages 99–108. DOI: 10.1145/2523649.2523658.

[102] RW van der Heijden, S Dietzel, and F Kargl. "Misbehavior Detection in Vehicular Ad-hoc Networks". In: *Proceedings of the 1st GI/ITG KuVS Fachgespräch Inter-Vehicle Communication*. FG-IVC. 2013.

[103] RW van der Heijden, S Dietzel, and F Kargl. "SeDyA: Secure Dynamic Aggregation in VANETs". In: *Proceedings of the sixth ACM conference on Security and privacy in wireless and mobile networks*. WiSec. ACM, April 2013, pages 131–142. DOI: 10.1145/2462096.2462119.

[104] IEEE. *IEEE 1609.2: IEEE Standard for Wireless Access in Vehicular Environments - Security Services for Applications and Management Messages*. IEEE Vehicular Technology Society, 2013.

[105] B Khaleghi, A Khamis, FO Karray, and SN Razavi. "Multisensor data fusion: A review of the state-of-the-art". In: *Information Fusion* 14.1 (January 2013), pages 28–44. DOI: 10.1016/j.inffus.2011.08.001.

[106] National Institute of Standards and Technology. *Digital Signature Standard*. Technical report FIPS PUB 186-4. Federal Information Processing Standards, July 2013.

[107] J Ploeg, E Semsar-Kazerooni, G Lijster, N van de Wouw, and H Nijmeijer. "Graceful degradation of CACC performance subject to unreliable wireless communication". In: *Intl. IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*. November 2013, pages 1210–1216. DOI: 10.1109/ITSC.2013.6728397.

[108] H Stübing. "Multilayered Security and Privacy Protection in Car-to-X Networks". PhD thesis. 2013. DOI: 10.1007/978-3-658-02531-1.

[109] W Whyte, A Weimerskirch, V Kumar, and T Hehn. "A security credential management system for V2V communications". In: *Vehicular Networking Conference*. IEEE, December 2013, pages 1–8. DOI: 10.1109/VNC.2013.6737583.

[110] T Yang, W Xin, L Yu, Y Yang, J Hu, and Z Chen. "MisDis: An Efficent Misbehavior Discovering Method Based on Accountability and State Machine in VANET". In: *Lecture Notes in Computer Science: Asia-Pacific Web Conference 2013*. Edited by Y Ishikawa, J Li, W Wang, R Zhang, and W Zhang. Volume 7808. Springer, 2013, pages 583–594. DOI: 10.1007/978-3-642-37401-2_57.

[111] N Bißmeyer. "Misbehavior Detection and Attacker Identification in Vehicular Ad-hoc Networks". PhD thesis. Darmstadt Technical University, 2014. URL: http://tuprints.ulb.tu-darmstadt.de/4257/.

[112] M Dacier, F Kargl, H König, and A Valdes. "Network Attack Detection and Defense: Securing Industrial Control Systems for Critical Infrastructures (Dagstuhl Seminar 14292)". In: *Dagstuhl Reports* 4.7 (2014). Edited by M Dacier, F Kargl, H König, A Valdes, and RW van der Heijden, pages 62–79. DOI: 10.4230/DagRep.4.7.62.

[113] J Dezert, A Tchamova, D Han, and JM Tacnet. "Can we trust subjective logic for information fusion?" In: *17th International Conference on Information Fusion (FUSION)*. IEEE. 2014, pages 1–8.

[114] S Dietzel, J Gürtler, RW van der Heijden, and F Kargl. "Redundancy-based Statistical Analysis for Insider Attack Detection in VANET Aggregation Schemes". In: *Vehicular Networking Conference*. VNC. IEEE, December 2014, pages 135–142. DOI: 10.1109/VNC.2014.7013332.

[115] S Dietzel, RW van der Heijden, H Decke, and F Kargl. "A Flexible, Subjective Logic-based Framework for Misbehavior Detection in V2V Networks". In: *15th International Symposium on a World of Wireless, Mobile and Multimedia Networks*. WoWMoM. IEEE, June 2014, pages 1–6. DOI: 10.1109/WoWMoM.2014.6918989.

[116] F Engelmann, T Lukaseder, B Erb, RW van der Heijden, and F Kargl. "Dynamic Packet-Filtering in High-Speed Networks using NetFPGAs". In: *Third International Conference on Future Generation Communication Technologies*. FGCT. Best paper award. IEEE, August 2014, pages 55–59. DOI: 10.1109/FGCT.2014.6933224.

[117] RW van der Heijden and F Kargl. "Open Issues in Data-Centric Misbehavior Detection for VANETs". In: *Proceedings of the 2nd GI/ITG KuVS Fachgespräch Inter-Vehicle Communication*. FG-IVC. University of Luxembourg, Interdisciplinary Centre for Security, Reliability and Trust (SnT), 2014, pages 24–26.

[118] F Kargl, RW van der Heijden, H König, A Valdes, and M Dacier. "Dagstuhl Manifesto (Network Attack Detection and Defense: Securing Industrial Control Systems for Critical Infrastructures)". In: *Informatik Spektrum* 47.6 (2014), pages 605–607.

[119] F Kargl, RW van der Heijden, H König, A Valdes, and M Dacier. "Insights on the Security and Dependability of Industrial Control Systems". In: *IEEE Security & Privacy* 12.6 (November 2014), pages 75–78. DOI: 10.1109/MSP.2014.120.

[120] R Mitchell and IR Chen. "A Survey of Intrusion Detection Techniques for Cyber-physical Systems". In: *ACM Comput. Surv.* 46.4 (March 2014), 55:1–55:29. DOI: 10.1145/2542049.

[121] M Segata, B Bloessl, S Joerer, F Dressler, and RL Cigno. "Supporting platooning maneuvers through IVC: An initial protocol analysis for the JOIN maneuver". In: *Proceedings of the 11th Annual Conference on Wireless On-demand Network Systems and Services*. WONS. April 2014, pages 130–137. DOI: 10.1109/WONS.2014.6814733.

[122] M Segata, S Joerer, B Bloessl, C Sommer, F Dressler, and R Lo Cigno. "PLEXE: A Platooning Extension for Veins". In: *6th IEEE Vehicular Networking Conference (VNC 2014)*. Paderborn, Germany: IEEE, December 2014, pages 53–60. DOI: 10.1109/VNC.2014.7013309.

[123] M Amoozadeh, A Raghuramu, C n. Chuah, D Ghosal, HM Zhang, J Rowe, and K Levitt. "Security vulnerabilities of connected vehicle streams and their impact on cooperative driving". In: *IEEE Communications Magazine* 53.6 (June 2015), pages 126–132. DOI: 10.1109/MCOM.2015.7120028.

[124] M di Bernardo, A Salvi, and S Santini. "Distributed Consensus Strategy for Platooning of Vehicles in the Presence of Time-Varying Heterogeneous Communication Delays". In: *IEEE Transactions on Intelligent Transportation Systems* 16.1 (February 2015), pages 102–112. DOI: 10.1109/TITS.2014.2328439.

[125] L Codeca, R Frank, and T Engel. "Luxembourg SUMO Traffic (LuST) Scenario: 24 hours of mobility for vehicular networking research". In: *Vehicular Networking Conference (VNC), 2015 IEEE*. December 2015, pages 1–8. DOI: 10.1109/VNC.2015.7385539.

[126] S Dadras, RM Gerdes, and R Sharma. "Vehicular Platooning in an Adversarial Environment". In: *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*. ASIA CCS '15. Singapore, Republic of Singapore: ACM, 2015, pages 167–178. DOI: 10.1145/2714576.2714619.

[127] B DeBruhl, S Weerakkody, B Sinopoli, and P Tague. "Is Your Commute Driving You Crazy?: A Study of Misbehavior in Vehicular Platoons". In: *Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. WiSec '15. New York, New York: ACM, 2015, 22:1–22:11. DOI: 10.1145/2766498.2766505.

[128] S Dietzel. "Resilient In-Network Aggregation for Vehicular Networks". PhD thesis. University of Twente, April 2015. DOI: 10.3990/1.9789036538527.

[129] S Dietzel, RW van der Heijden, J Petit, and F Kargl. "Context-Adaptive Detection of Insider Attacks in VANET Information Dissemination Schemes". In: *Vehicular Networking Conference*. VNC. IEEE, December 2015, pages 287–294. DOI: 10.1109/VNC.2015.7385590.

[130] M Fazouane, H Kopp, RW van der Heijden, D Le Métayer, and F Kargl. "Formal Verification of Privacy Properties in Electric Vehicle Charging". In: *Engineering Secure Software and Systems*. Edited by F Piessens, J Caballero, and N Bielova. Cham: Springer International Publishing, 2015, pages 17–33. DOI: 10.1007/978-3-319-15618-7_2.

[131] S Gisdakis, T Giannetsos, and P Papadimitratos. "SHIELD: A Data Verification Framework for Participatory Sensing Systems". In: *ACM Conference on Security & Privacy in Wireless and Mobile Networks (ACM WiSec)*. New York, NY, USA, June 2015, 16:1–16:12. DOI: 10.1145/2766498.2766503.

[132] S Kleber, RW van der Heijden, H Kopp, and F Kargl. "Terrorist Fraud Resistance of Distance Bounding Protocols Employing Physical Unclonable Functions". In: *2015 International Conference and Workshops on Networked Systems*. NetSys. IEEE, March 2015, pages 1–8. DOI: 10.1109/NetSys.2015.7089068.

[133] E Koutrouli and A Tsalgatidou. "Reputation Systems Evaluation Survey". In: *ACM Comput. Surv.* 48.3 (December 2015), 35:1–35:28. DOI: 10.1145/2835373.

[134] J Petit, F Schaub, M Feiri, and F Kargl. "Pseudonym Schemes in Vehicular Networks: A Survey". In: *Communications Surveys Tutorials, IEEE* 17.1 (2015), pages 228–255. DOI: 10.1109/COMST.2014.2345420.

[135] J Petit and S Shladover. "Potential Cyberattacks on Automated Vehicles". In: *Intelligent Transportation Systems, IEEE Transactions on* 16.2 (April 2015), pages 546–556. DOI: 10.1109/TITS.2014.2342271.

[136] M Saini, A Alelaiwi, and AE Saddik. "How Close Are We to Realizing a Pragmatic VANET Solution? A Meta-Survey". In: *ACM Comput. Surv.* 48.2 (November 2015), 29:1–29:40. DOI: 10.1145/2817552.

[137] T Saito and M Rehmsmeier. "The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets". In: *PLOS ONE* 10.3 (March 2015), pages 1–21. DOI: 10.1371/journal.pone.0118432.

[138] C Sommer, J Härri, F Hrizi, B Schünemann, and F Dressler. "Simulation Tools and Techniques for Vehicular Communications and Applications". In: *Vehicular ad hoc Networks*. Edited by C Campolo, A Molinaro, and R Scopigno. Springer International Publishing, 2015, pages 365–392. DOI: 10.1007/978-3-319-15497-8_13.

[139] A Al-Momani, RW van der Heijden, F Kargl, and C Waldschmidt. "Exploiting Propagation Effects for Authentication and Misbehavior Detection in VANETs". In: *Vehicular Networking Conference*. VNC. IEEE, December 2016, pages 1–4. DOI: 10.1109/VNC.2016.7835973.

[140] N Dreyer, A Moller, ZH Mir, F Filali, and T Kurner. "A Data Traffic Steering Algorithm for IEEE 802.11p/LTE Hybrid Vehicular Networks". In: *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*. September 2016, pages 1–6. DOI: 10.1109/VTCFall.2016.7880850.

[141] RW van der Heijden, A Al-Momani, F Kargl, and OMF Abu-Sharkh. "Enhanced Position Verification for VANETs using Subjective Logic". In: *Proceedings of the 84th Vehicular Technology Conference*. VTC-Fall. IEEE, September 2016. URL: http://arxiv.org/abs/1703.10399.

[142] RW van der Heijden, S Dietzel, T Leinmüller, and F Kargl. *Survey on Misbehavior Detection in Cooperative Intelligent Transportation Systems*. arXiv preprint. 2016. URL: https://arxiv.org/abs/1610.06810.

[143] D Jia, K Lu, J Wang, X Zhang, and X Shen. "A Survey on Platoon-Based Vehicular Cyber-Physical Systems". In: *IEEE Communications Surveys & Tutorials* 18.1 (2016), pages 263–284. DOI: 10.1109/COMST.2015.2410831.

[144] A Jøsang. *Subjective Logic: A Formalism for Reasoning Under Uncertainty*. Artificial Intelligence: Foundations, Theory and Algorithms. Springer International Publishing Switzerland, 2016. DOI: 10.1007/978-3-319-42337-1.

[145] T Lukaseder, L Bradatsch, B Erb, RW van der Heijden, and F Kargl. "A Comparison of TCP Congestion Control Algorithms in 10G Networks". In: *41st Conference on Local Computer Networks*. LCN. IEEE, November 2016, pages 706–714. DOI: 10.1109/LCN.2016.121.

[146] D Meißner, B Erb, RW van der Heijden, K Lange, and F Kargl. "Mobile Triage Management in Disaster Area Networks Using Decentralized Replication". In: *Proceedings of the Eleventh ACM Workshop on Challenged Networks*. CHANTS. ACM, 2016, pages 7–12. DOI: 10.1145/2979683.2979689.

[147] D Schmidt, K Radke, S Camtepe, E Foo, and M Ren. "A Survey and Analysis of the GNSS Spoofing Threat and Countermeasures". In: *ACM Comput. Surv.* 48.4 (May 2016), 64:1–64:31. DOI: 10.1145/2897166.

[148] G Shafer. "Dempster's rule of combination". In: *International Journal of Approximate Reasoning* 79 (2016). 40 years of Research on Dempster-Shafer theory, pages 26–40. DOI: https://doi.org/10.1016/j.ijar.2015.12.009.

[149] K Zaidi, MB Milojevic, V Rakocevic, A Nallanathan, and M Rajarajan. "Host-Based Intrusion Detection for VANETs: A Statistical Approach to Rogue Node Detection". In: *Transactions on Vehicular Technology* 65.8 (August 2016), pages 6703–6714. DOI: 10.1109/TVT.2015.2480244.

[150] A Alipour-Fanid, M Dabaghchian, H Zhang, and K Zeng. "String Stability Analysis of Cooperative Adaptive Cruise Control under Jamming Attacks". In: *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*. January 2017, pages 157–162. DOI: 10.1109/HASE.2017.39.

[151] M Charitos and G Kalivas. "MIMO HetNet IEEE 802.11p–LTE deployment in a vehicular urban environment". In: *Vehicular Communications* 9 (July 2017), pages 222–232. DOI: 10.1016/j.vehcom.2016.12.004.

[152] F Diemer. "Improving position verification in VANETs". Bachelor Thesis in cooperation with Rens W. van der Heijden. Ulm University, 2017.

[153] S Eberz, KB Rasmussen, V Lenders, and I Martinovic. "Evaluating Behavioral Biometrics for Continuous Authentication: Challenges and Metrics". In: *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. ASIA CCS '17. Abu Dhabi, United Arab Emirates: ACM, 2017, pages 386–399. DOI: 10.1145/3052973.3053032.

[154] FA Ghaleb, A Zainal, MA Rassam, and F Mohammed. "An effective misbehavior detection model using artificial neural network for vehicular ad hoc network applications". In: *2017 IEEE Conference on Application, Information and Network Security (AINS)*. November 2017, pages 13–18. DOI: 10.1109/AINS.2017.8270417.

[155] J Giraldo, E Sarkar, AA Cardenas, M Maniatakos, and M Kantarcioglu. "Security and Privacy in Cyber-Physical Systems: A Survey of Surveys". In: *IEEE Design Test* 34.4 (August 2017), pages 7–17. DOI: 10.1109/MDAT.2017.2709310.

[156] H Hasrouny, AE Samhat, C Bassil, and A Laouiti. "VANet security challenges and solutions: A survey". In: *Vehicular Communications* 7.Supplement C (January 2017), pages 7–20. DOI: 10.1016/j.vehcom.2017.01.002.

[157] RW van der Heijden, F Engelmann, D Mödinger, F Schönig, and F Kargl. "Blackchain: Scalability for Resource-Constrained Accountable Vehicle-to-X Communication". In: *Proceedings of the 1st Workshop on Scalable and Resilient Infrastructures for Distributed Ledgers*. SERIAL. ACM, 2017. DOI: 10.1145/3152824.3152828.

[158] RW van der Heijden and F Kargl. "Evaluating Misbehavior Detection for Vehicular Networks". In: *Proceedings of the 5th GI/ITG KuVS Fachgespräch Inter-Vehicle Communication*. FG-IVC. Friedrich-Alexander-Universität Erlangen-Nürnberg, Department of Computer Science, 2017, pages 5–8.

[159] RW van der Heijden, T Lukaseder, and F Kargl. "Analyzing Attacks on Cooperative Adaptive Cruise Control (CACC)". In: *Vehicular Networking Conference*. VNC. Best paper award. IEEE, November 2017, pages 45–52. DOI: 10.1109/VNC.2017.8275598.

[160] A Jøsang, D Wang, and J Zhang. "Multi-source fusion in subjective logic". In: *2017 20th International Conference on Information Fusion (Fusion)*. IEEE, July 2017, pages 1–8. DOI: 10.23919/ICIF.2017.8009820.

[161] A Kumar, S Mehta, and D Vijaykeerthy. "An Introduction to Adversarial Machine Learning". In: *Big Data Analytics*. Edited by PK Reddy, A Sureka, S Chakravarthy, and S Bhalla. Cham: Springer International Publishing, 2017, pages 293–299. DOI: 10.1007/978-3-319-72413-3_20.

[162] T Lukaseder, A Hunt, C Stehle, D Wagner, RW van der Heijden, and F Kargl. "An Extensible Host-Agnostic Framework for SDN-Assisted DDoS-Mitigation". In: *42nd Conference on Local Computer Networks*. LCN. IEEE, October 2017, pages 619–622. DOI: 10.1109/LCN.2017.103.

[163] R Molina-Masegosa and J Gozalvez. "LTE-V for Sidelink 5G V2X Vehicular Communications: A New 5G Technology for Short-Range Vehicle-to-Everything Communications". In: *IEEE Vehicular Technology Magazine* 12.4 (December 2017), pages 30–39. DOI: 10.1109/MVT.2017.2752798.

[164] F Sakiz and S Sen. "A survey of attacks and detection mechanisms on intelligent transportation systems: VANETs and IoV". In: *Ad Hoc Networks* 61.Supplement C (June 2017), pages 33–50. DOI: 10.1016/j.adhoc.2017.03.006.

[165] M Sun, M Li, and R Gerdes. "A data trust framework for VANETs enabling false data detection and secure vehicle tracking". In: *2017 IEEE Conference on Communications and Network Security (CNS)*. October 2017, pages 1–9. DOI: 10.1109/CNS.2017.8228654.

[166] J Whitefield, L Chen, T Giannetsos, S Schneider, and H Treharne. "Privacy-enhanced capabilities for VANETs using direct anonymous attestation". In: *2017 IEEE Vehicular Networking Conference (VNC)*. November 2017, pages 123–130. DOI: 10.1109/VNC.2017.8275615.

[167] C Yavvari, Z Duric, and D Wijesekera. "Vehicular dynamics based plausibility checking". In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. October 2017, pages 1–8. DOI: 10.1109/ITSC.2017.8317883.

[168] B Brecht, D Therriault, A Weimerskirch, W Whyte, V Kumar, T Hehn, and R Goudy. "A Security Credential Management System for V2X Communications". In: *IEEE Transactions on Intelligent Transportation Systems* (2018), pages 1–22. DOI: 10.1109/TITS.2018.2797529.

[169] K Eykholt, I Evtimov, E Fernandes, B Li, A Rahmati, C Xiao, A Prakash, T Kohno, and D Song. "Robust Physical-World Attacks on Deep Learning Models". In: *Proceedings of the Conference on Computer Vision and Pattern Recognition*. arXiv: 1707.08945. 2018. URL: http://arxiv.org/abs/1707.08945 (visited on 07/03/2018).

[170] RW van der Heijden, S Dietzel, T Leinmüller, and F Kargl. "Survey on Misbehavior Detection in Cooperative Intelligent Transportation Systems". In: *IEEE Communication Surveys & Tutorials* (2018). DOI: 10.1109/COMST.2018.2873088.

[171] RW van der Heijden, H Kopp, and F Kargl. "Multi-Source Fusion Operations in Subjective Logic". In: *International Conference on Information Fusion*. FUSION. IEEE, July 2018. URL: https://arxiv.org/abs/1805.01388.

[172] RW van der Heijden, T Lukaseder, and F Kargl. "VeReMi: A Dataset for Comparable Evaluation of Misbehavior Detection in VANETs". In: *Proceedings of the 14th EAI International Conference on Security and Privacy in Communication Networks*. SecureComm. Springer, August 2018. URL: https://arxiv.org/abs/1804.06701.

[173] M Khodaei, H Jin, and P Papadimitratos. "SECMACE: Scalable and Robust Identity and Credential Management Infrastructure in Vehicular Communication Systems". In: *IEEE Transactions on Intelligent Transportation Systems (IEEE ITS)* (April 2018). DOI: 10.1109/TITS.2017.2722688.

[174] M Matousek, M Yassin, A Al-Momani, RW van der Heijden, and F Kargl. "Robust Detection of Anomalous Driving Behavior". In: *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*. June 2018, pages 1–5. DOI: 10.1109/VTCSpring.2018.8417777.

[175] E Uhlemann. "The Battle of Technologies or the Battle of Business Models? [Connected Vehicles]". In: *IEEE Vehicular Technology Magazine* 13.1 (March 2018), pages 14–18. DOI: 10.1109/MVT.2017.2781539.

[176] Y Yao, B Xiao, G Wu, X Liu, Z Yu, K Zhang, and X Zhou. "Multi-channel based Sybil Attack Detection in Vehicular Ad Hoc Networks using RSSI". In: *IEEE Transactions on Mobile Computing* (2018), pages 1–1. DOI: 10.1109/TMC.2018.2833849.

[177] C Damgaard. *Gini Coefficient*. From MathWord – A Wolfram Web Resource, created by Eric W. Weisstein. http://mathworld.wolfram.com/GiniCoefficient.html (Accessed February 9th, 2018).