Survey on Misbehavior Detection in Cooperative Intelligent Transportation Systems*

Rens W. van der Heijden † Stefan Dietzel ‡ Tim Leinmüller § Frank Kargl †

October 24, 2016

Abstract

Cooperative Intelligent Transportation Systems (cITS) are a promising technology to enhance driving safety and efficiency. Vehicles communicate wirelessly with other vehicles and infrastructure, thereby creating a highly dynamic and heterogeneously managed ad-hoc network. It is these network properties that make it a challenging task to protect integrity of the data and guarantee its correctness. A major component is the problem that traditional security mechanisms like PKI-based asymmetric cryptography only exclude outsider attackers that do not possess key material. However, because attackers can be insiders within the network (i.e., possess valid key material), this approach cannot detect all possible attacks. In this survey, we present misbehavior detection mechanisms that can detect such insider attacks based on attacker behavior and information analysis. In contrast to well-known intrusion detection for classical IT systems, these misbehavior detection mechanisms analyze information semantics to detect attacks, which aligns better with highly application-tailored communication protocols foreseen for cITS. In our survey, we provide an extensive introduction to the cITS ecosystem and discuss shortcomings of PKI-based security. We derive and discuss a classification for misbehavior detection mechanisms, provide an in-depth overview of seminal papers on the topic, and highlight open issues and possible future research trends.

1 Introduction

Throughout the field of computer science, securing systems against malicious attackers has become a fundamental requirement for safe and dependable operation of all kinds of applications. Today, professional attacks against systems, which are mounted by large criminal organizations or even governments get more and more common. At the same time, computer systems get more and more intertwined with the real world. The term cyber-physical systems (CPS) has been coined to encompass all kinds of systems that are characterized by a large deployment of networked devices equipped with both sensors and actuators. Unlike traditional embedded systems, where individual nodes perform interactions with the real world in strongly constrained environments, CPS are highly networked, deployed in large regions, and may contain nodes with high computational power. Notably, the content transferred in these networks is highly predictable, and relates directly to some real-world phenomenon [1].

One example of such a system is a cooperative intelligent transportation system (cITS), which consists of vehicles, road-side units and back-end systems, and is the main focus of this survey. However, this survey will also discuss the applicability of techniques from cITS to other notable examples of CPS, such as smart grids and industrial control systems (ICS).

Foreseen benefits of cITS include improved road-safety; enhanced driving experience and greener driving using improved traffic management; and providing infotainment services, such as Internet access, chat, and

*This article is a pre-print of a survey article that will be submitted in the future.

†Ulm University, Germany

‡Humboldt-Universität, Germany

§DENSO Automotive Deutschland GmbH, Germany

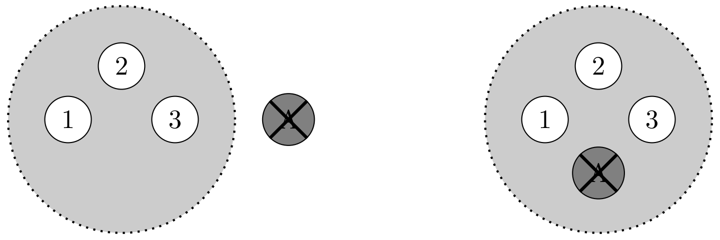


Figure 1: Proactive security is an important countermeasure against arbitrary attacks using commodity hardware, but it cannot protect against capable insider attackers.

multimedia streaming. Entertainment applications aside, the characterizing communication paradigm of all these applications is that sensors are used to measure real world conditions, which are then communicated over a ubiquitous network. This network is built up by equipping each vehicle with a wireless interface, and creating a dynamic ad-hoc network that can be accessed without further overhead. Vehicles use this network to send and receive information derived from their sensors and build a world model from received messages and local information, which they then analyze and act upon. The world model can be regarded as a knowledge base storing information about the real world. For example, one application called lane merging assistant, which increases safety by notifying the driver when a lane merge can be performed safely. This type of application relies on the exact positions of neighboring vehicles, exchanged over the network and stored in a world model.

1.1 Security in cITS

Being built into vehicles, cITS directly interact with humans. As a consequence, attacks have the potential to threaten the safety of people. For this reason, it is important to guarantee the integrity and availability of these systems, including the information they generate and operate on. A wide body of work is aiming to provide such protection, which can broadly categorized into **proactive** and **reactive** mechanisms [55], as shown in Figure 1. 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 access restriction using credentials that only valid users possess. For instance, in cITS, the typical approach is to use public key infrastructures (PKIs) and only hand out key material and certificates to vehicles and other authorized entities. All unauthorized entities are excluded from the system, because their messages do not contain valid signatures, which causes these messages to be rejected by authorized entities. This creates a trusted perimeter that encompasses all authorized entities, which requires potential attackers to possess or compromise valid credentials before they can access the system, reducing the number of attack vectors.

However, proactive security mechanisms cannot help to prevent or detect all attacks. In particular, we identify the following key characteristics of CPS, which determine their different security requirements.

Ubiquitous access In many CPS, there is either no clear boundary between insiders and outsiders, or this boundary contains so many devices that some of these may be subject to compromise. cITS are a prime example, as such networks are cooperatively formed by vehicles and road-side equipment.

Limited physical security As nodes in CPS are often distributed in a potentially hostile environment, they may be subject to hijacking, analysis, and reprogramming by attackers, even if they are centrally managed. In cITS, the network nodes may even be under the control of malicious participants, because attackers with sufficient funds can buy vehicles to modify them and mount attacks.

Potential impact CPS control critical infrastructures in which a large variety of attackers may have an interest. Due to the direct connection with human safety, CPS can be a target for attackers with high financial resources. In cITS, safety applications are designed to improve road safety, which can be negatively impacted if these applications are attacked successfully.

Sensor values as security assets In typical network security scenarios, the primary security assets are the components in the network and the packets or user data transmitted in the network; these assets may be subject to attacks, and the goal in network security is to protect them from these attacks. On the contrary, in CPS, the primary security assets are the sensor values contained within transmitted packets, as well as the actuators controlled based on this data. Consequently, the networking patterns in CPS are application-dependent and built around the semantics of exchanged information, rather than the classical ISO/OSI stack of network layers. This focus on semantics strongly motivates the design and use of misbehavior detection to detect attacks.

Even when considering other proactive security measures, such as Proof-of-Work (PoW) mechanisms [62], the best result for these characteristics is to hinder an attacker. For this reason, proactive mechanisms have to be complemented by **reactive security**. Reactive security is provided by all kinds of mechanisms that analyze system behavior or state to detect attacks and failures. In the traditional Internet, intrusion detection systems (IDS) and intrusion prevention systems (IPS) are examples for reactive security mechanisms. Both areas are actively researched and many surveys on the topics exist [80, 49, 21, 16].

Intrusion detection has been based on many different approaches; one common classification is to distinguish between detection of known attack patterns (signature-based) and detection of anomalous behavior (anomaly-based). Important to note is that these are both dependent on the representation of each message: a signature-based IDS may detect by matching known malicious strings against packets; if a match is found, an attack is detected. Similarly, anomaly-based IDS work by analyzing a message as compared to other messages in the network; if the message is sufficiently different from *normal* traffic, an attack is detected. Unlike these approaches, we define misbehavior detection as using (application) semantics of the transmitted data to complement detection, which reflects the essential characteristics of the term as it is used with respect to routing attacks on ad-hoc networks in earlier literature [59]. For example, verifying whether a warning message is consistent with the perceived state of the observed system is misbehavior detection, while checking the protocol-semantic correctness of a TCP header is intrusion detection. If this warning message is not consistent, an attack is detected – an attacker attempted to issue a false warning. The third method for intrusion detection mentioned in the taxonomy of [80] is similar; stateful protocol, or specification-based detection. In this method, the protocol specification is used to analyze whether the received messages correspond to the protocol. However, note that this only works for stateful protocols; in CPS, in particular in cITS, most protocols are not stateful. We argue misbehavior detection is a forth method, which uses application state to perform detection of malicious messages.

Some misbehavior detection mechanisms also has some similarities to the concept of Byzantine fault tolerance (BFT) in distributed systems, in particular in those cases where some notion of consensus is required between the nodes in the network [12]. In particular, BFT tells us that given a system tolerating f faults, at least $2f + 1$ correct nodes are required, such that we will always have a Byzantine quorum. A Byzantine quorum is any set of at least $\frac{N+f+1}{2}$ nodes, for a network of N noes. As a result, any two Byzantine quorums will intersect in at least one node. As a result, when a Byzantine quorum for a particular result is reached, we can be certain that this result is the correct outcome of the algorithm executed by all correct nodes. However, a major difference between typical distributed systems and many types of CPS, such as cITS, the amount of nodes N is not known, or only a small subset of these nodes are actually in communication range. Another important difference is the fact that connectivity may be too sparse, meaning the necessary quorum may not be reached fast enough.

In this article, we provide a comprehensive survey of different misbehavior detection mechanisms; we choose cITS as our prime application domain. Due to the challenging network topology and application-specific networking paradigms applied in cITS, we believe that misbehavior detection mechanisms for these systems offer interesting properties that can be applied to a wide range of use cases, including other CPS.

Our main contributions can be summarized as follows:

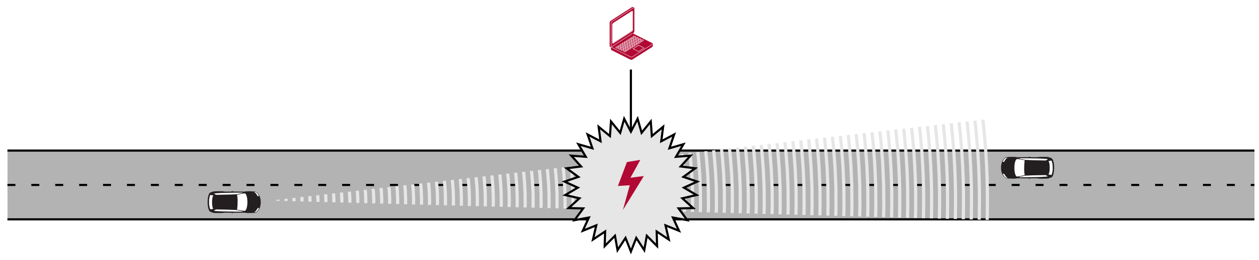
- In the remainder of this section, we provide an overview of cITS, cITS security and misbehavior; the system model, current standardization, and security approaches are discussed in more detail in Section 2.
- We define misbehavior, an attacker model, and provide a taxonomy for misbehavior detection in cITS in Section 3.
- We provide an extensive overview of seminal works in different areas of misbehavior detection in Section 4.
- We discuss solved and open challenges (Section 5) and point out commonalities and differences of cITS misbehavior detection and misbehavior detection in other domains (Section 5.3).

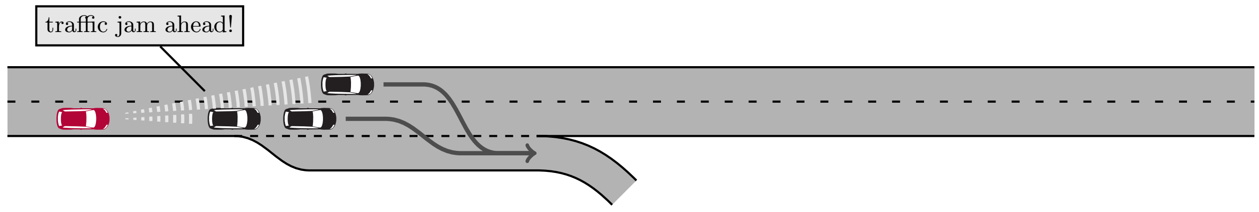
1.2 cITS overview

Depending on the application type, different communication patterns [84] are used in cITS environments, which will be surveyed in more detail in Section 2. For active safety applications, which allow the vehicle to react directly to dangerous situations, communication is predominantly between vehicles (V2V); often broadcast of messages to neighbors in direct communication range suffices to provide applications with enough information. A core service is the so-called beaconing, a high frequency ($\approx$ 1–10 Hz) broadcast of up-to-date status information like current position, time, speed, heading, and other vehicle data to 1-hop communication neighbors. Beaconing is standardized in Europe by ETSI as the cooperative awareness message (CAM) [24]. For applications requiring dissemination of messages in a wider area, ETSI foresees so-called decentralized environmental notification message (DENM) [22]. These messages may be forwarded over multiple hops to inform vehicles further away about important events, such as approaching emergency vehicles. In contrast to CAMs, DENMs are event-triggered and not periodic.

Although V2V communication, that is communication purely between vehicles, is able to support a large number of use cases, there is a requirement to complement V2V communication with vehicle-to-infrastructure (V2I) communication. Infrastructure can help to increase communication range during the initial deployment phase – consider, for instance, roadside units at important intersections –, and provide communication links to back-end infrastructure and the Internet. Back-end connections can be used for infotainment services but also to provide over-the-air software updates or update security credentials stored in vehicles. However, it is unlikely that roadside units will be deployed on a large scale due to of estimated deployment and operation costs of 3,000–5,000 US dollars per road side unit (RSU) [3].

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 universal mobile telecommunications system (UMTS) have already been deployed on a larger scale. However, UMTS deployment is far from complete even in some urban areas, and it is debatable whether cellular networks could scale to support large-scale vehicular networks due to the breathing cell problem [89], which reduces coverage when many participants use the network. Moreover, cellular support will likely remain restricted to more expensive vehicles as long as cellular usage still incurs considerable fees. Hence, we will focus on V2V and V2I communication and supporting infrastructure in the following. Note that unlike UMTS, RSUs actually participate in the vehicular network, and are capable of transmitting and processing messages directly in the network, rather than providing Internet access to vehicles. In particular, this means that such devices can also be used to perform application-specific tasks (e.g., there are works that use RSUs as an additional authority to verify participants [14]).


(a) Jamming attack



(b) Fake message injection attack

Figure 2: Two examples for different types of misbehavior.

1.3 Impact 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 3. For now, consider two example attacks to demonstrate the range of possibilities. Figure 2a shows a jamming attack. Here, an attacker creates noise on the wireless channel, which hinders the message transfer between two benign vehicles. To mount this attack, the attacker does not need any particular knowledge about the semantics of the messages that the vehicles exchange. It is enough for the attacker to know the specification of the wireless communication channel, which is publicly available in standard documents.

Figure 2b 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 (nonexistent) 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 inexistent 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.

If we consider safety applications or automated driving [68], it is conceivable that attacks on intelligent transportation systems can even lead to accidents, possibly threatening lives of passengers. For instance, an attacker could inject a fake message that warns approaching vehicles about an obstacle on the road. Receiving vehicles will break hard to avoid the obstacle. But vehicles that do not receive the message will be unaware and might not be able to react in time.

1.4 cITS Security

In order to mitigate possible attack vectors, security features are actively considered by both academia and industry. The basic approach is to set up a PKIs and provide each vehicle with an asymmetric key pair, of which the public key is amended with a number of attributes (e.g., lifetime, vehicle type, license plate number) and signed by the key issuing authority [72, 47, 63]. The signed public key and attributes together are known as the vehicle's certificate, because they certify that the key pair was issued by a trusted authority and that the authority vouches for the additionally signed attributes. Because of the contained unique identifiers, such as the vehicle's license plate number, the certificate acts as a long-term identifier of a vehicle.

Each outgoing message is signed using the sender's secret key, and the resulting signature, as well as the sender's certificate are attached to the message. Attaching the certificate is necessary, because receivers will likely neither have interacted with the sender earlier nor can obtain the certificate from a key server on the fly. Receiving vehicles first check the signature for correctness and then check the certificate. If both are correct, receivers can distinguish messages from actual vehicles from arbitrary messages that were generated using commodity hardware and injected into the network. To improve privacy, long-term certificates can be replaced by short-term identifiers, so-called pseudonyms, which still make sure that messages originate from vehicles, but cannot be easily used to gather location traces of specific vehicles [47, 63, 67]. More details regarding certificates and pseudonyms will be discussed in Sections 2.3 and 2.4.

While signature and certificates effectively thwart most attacks using commodity hardware, the question remains how effective they are against attackers that 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 can easily extract it, transfer it to other devices, and create arbitrary messages that are correctly signed. Therefore, trusted hardware, usually in the form of a hardware security module, is considered to protect key material [72, 47]. The idea is to store (secret) key material within a tamper resistant component that is protected against all access from outside. 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. The certificates associated with discarded communication units can be revoked, which provides a degree of protection against extracting old key material.

However, even if key material is kept in trusted hardware, the application code is still untrusted, and 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 [75]. While such an architecture might successfully hinder attackers from modifying messages, it comes with a number of drawbacks. First, running all software on trusted hardware will 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. Moreover, even if attackers are not able to modify the software, they might 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 CAN bus or directly modify sensor hardware. Therefore, it is likely that, although key material will be kept within trusted hardware, software running on the vehicle can be controlled by a capable attacker. In conclusion, we have to consider that messages with modified, fake content are circulating within the network.

2 System Model

In order to better understand the type of communication and interaction in cITS, we will provide an overview of the system model. First introducing application domains, and the network structure, we continue to explain communication and ongoing standardization. We focus our references to standardization on current progress in the EU but point out notable differences in US standardization where helpful.

2.1 Application Domains

Three broad application domains are road safety, traffic efficiency, and business and infotainment services. Safety applications can further be subdivided into active and passive safety applications. Passive safety applications raise driver awareness of the surroundings to help detecting obstacles and potentially dangerous situations earlier. An example passive safety application is a lane merging assistant. Here, drivers are informed of the exact vehicle positions on adjacent lanes to help the driver merge once a sufficiently large gap is available. Other examples for passive safety applications are severe braking warnings, weather warnings, and accident warnings. Whereas passive safety applications only inform drivers, active safety applications can directly interact with vehicle steering and acceleration. An active safety application could engage automatic breaking if an accident would be unavoidable otherwise. Due to liability issues, passive safety applications are in more active development currently.

Traffic efficiency applications aim to improve traffic flow by providing enhanced navigation services. Here, the impact are shorter travel times, more enjoyable driving experience, and less traffic jams. An example traffic efficiency application is an advanced traffic information system that informs all vehicles with a live approximation of their surrounding traffic situation to allow for dynamic rerouting. Similarly, information about free parking spaces can shorten search times.

Finally, infotainment applications provide value-added services to drivers. Examples are paid services such as video streaming, voice over IP services, or application downloads and updates. Contrary to safety and traffic efficiency applications, infotainment applications require mostly unicast routing for mobile nodes to be provided.

2.2 Network participants and communication

cITS are systems that consist of multiple vehicles that communicate with each other. The systems consists of so called ITS-stations (ITS-Ss), which can be vehicles or road side units (RSUs).

Vehicles are mobile nodes in cITS, equipped with an on-board unit (OBU), which sends and receives messages. The communication is based on IEEE 802.11. It was originally standardized in the amendment 802.11p and is now part of the 2012 revision of 802.11. In Europe, ITS-G5 is based on 802.11p and provides the basic communication primitives. Communication is performed at 5.9 GHz with a communication range of approximately 200 to 300 meters. The majority of network participants are mobile nodes. However, infrastructure can be used to provide more information and help applications. RSUs are infrastructure elements that send information about the current traffic situation, for instance, the status and duration of traffic signals.

All ITS-Ss are able to send and receive messages. In contrast to routing in classical IT systems such as the Internet, multicast and broadcast are the predominant networking patterns. The reasons are simple: most information originates form environment sensors, and the network's goal is to spread such information and increase surrounding awareness in larger areas. Almost all packets are relevant for many other vehicles and are therefore broadcast. Second, this allows the network to exploit the simple fact that wireless networks are a broadcast medium by nature. The group of vehicles that should receive messages can be expressed using relevance metrics. For instance, information about an accident is relevant for vehicles approaching the accident on the same or adjacent streets. To simplify communication patterns, relevance is often approximated by location: messages carry a region identifier, and the messages are disseminated within that region. Similarly, physical network topology replaces virtual topology as routing metric if messages need to be forwarded to specific vehicles.

To support such communication patterns, the European model currently foresees two types of messages originating from vehicles. The first one is a kind of beacon message, called cooperative awareness message (CAM) [24]. CAMs are sent in specific intervals, and the sending frequency varies from 1 to 10 Hz depending the application context.

CAMs consist of information that expresses properties of the sending vehicle or its sensed environment. This includes:

- station ID (unique identifier, maybe the pseudonym)
- position (longitude and latitude of global positioning system (GPS) position)
- heading
- speed
- acceleration

- steering angle

Supplemental information can be added, like vehicle role, vehicle dimensions, status of vehicle lights, path history and other information. CAMs are distributed as single hop broadcast message; these messages are not forwarded. Besides CAMs, the ETSI model foresees decentralized environmental notification messages (DENMs) [22][1]. DENMs are an 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 can be forwarded over multiple hops. As explained above, DENMs are usually relevant for specific geographical area. A geobroadcast protocol is specified [25], which first forwards messages to the designated target region and then broadcasts the message within that region.

With information from transmitted messages, every vehicle is able to build up a so-called local dynamic map (LDM). This is an internal storage of an ITS-S that contains information about the surrounding ITS-Ss, like vehicles and RSUs (such as traffic lights). The LDM serves as the central information database for applications.

Standardization in the US foresees messages similar to Europe's CAM and DENM. However, the US model only standardizes one type of message, the basic safety message (BSM) [60, 44]. BSMs can be divided into two parts. Part one (BSM.1) is similar to the European CAM. It consists of core elements like position, heading, speed, acceleration, steering wheel angle and vehicle size. BSMs can be sent at a frequency of up to 10 Hz. Part two of the message format (BSM.2) contains optional data elements. These elements are selected and sent on a triggered event, such as the activation of the vehicles' anti-lock braking system (ABS). These data elements are added to the BSM.1. The full BSM (consisting of BSM.1 and BSM.2) can be sent with a lower frequency to save bandwidth. In terms of message intent, the full BSM is comparable with DENMs in the European model, except that the BSM is a broadcast message: multihop communication is not foreseen in this model.

2.3 Security

Realizing the importance of safety-relevant applications, security measures have been considered early on during cITS standardization. Since most messages contain environmental information, which is intended for all interested network participants, confidentiality can be neglected. However, it is important to ensure message integrity to hinder unauthorized entities that want to interfere with proper system operation. Authorized entities are all vehicles, RSUs, and other infrastructure that is part of the network. Examples for unauthorized entities are attackers that use commodity laptops to inject messages into the network. All such unauthorized entities should be excluded from the network to implement proactive security.

To achieve this goal, certificates are used to ensure that vehicles can verify that information originates from other (trusted) vehicles. Each receiver of messages should be able to check if messages originate from authenticated and authorized members of the cITS. Additionally, it is important to detect manipulation of the transmitted messages. Thus, messages are secured using signatures and certificates to provide authenticity and integrity.

To implement integrity protection, a PKI is foreseen that is operated by government agencies, existing commercial PKI entities or other trustworthy parties. Every authentic and authorized member of the cITS receives a certificate from the certificate authority of the PKI. This certificate is based on the ETSI TS 103 097 standard [26] in Europe and the IEEE 1609.2 standard [43] in the US. The certificate consists of an identifier, certain vehicle properties, the vehicle role, the public key and a signature of a certification authority (CA). Every message is protected by a signature, which is created with the private key corresponding to the public key in the certificate. Using the certificate and public key, which are attached to all outgoing messages, receivers can verify the authenticity and the integrity of the transmitted message. In both European and American standards, the cryptographic algorithm is elliptic curve digital signature algorithm (ECDSA), using the p-224 or p-256 curve standardized by NIST [61]. For efficiency reasons, some authors have proposed certificates could be omitted periodically to save bandwidth [27]. Although this is very effective, because the

[1]There is on-going standardization of other message types, such as the signal phase and time (SPaT) and Topology Specification (TOPO) message types

certificate is the majority of the message payload, it leads to additional delays and messages of uncertain origin that may have been manipulated.

2.4 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. To prevent tracking of drivers and to protect their privacy, cITS, therefore, employs anonymization.

For this purpose, vehicles use so called pseudonym certificatess (PCs). Instead of using long term identities, messages are signed and transmitted with the PCs. The PCs have to change from time to time to prevent the tracking of the individual vehicle. For instance, a pseudonym can be used for one week, after which it is discarded. To further improve privacy, current standardization foresees a number of pseudonyms to be valid within the same time period. For instance, each vehicle could be able to use 20 pseudonyms per week, freely deciding which specific pseudonym to use at any time within that week.

After the PCs of a vehicle expire, it requests a new set from the certificate authority using its long term identity. This identity is used only to issue new PCs, and not for regular communication in the vehicular ad-hoc network (VANET).

While pseudonyms enhance privacy, their parallel use enables potential attacks, often referred to as Sybil attacks [20], which we will discuss extensively in this survey (in section 3.1.2), as it is one of the key challenges that misbehavior detection helps address.

3 Misbehavior and its detection

In this section, we will go into more detail concerning misbehavior and its detection. In particular, we discuss the attacker model, different types of realistic attackers and a taxonomy of misbehavior detection. As detection can be performed on different scopes, we also briefly address this topic and the closely related topics of misbehavior reporting and revocation. Finally, we will go into the topic of pseudonyms, in order to investigate the compatibility of the misbehavior detection mechanisms with this privacy-enhancing mechanism. We will use each of these aspects in our classification of the literature in the next section.

3.1 Definition and attacker model

In this subsection, we will treat the definition of misbehavior and the attacker model associated with it. We distinguish these two concepts because misbehavior gives a strong intuitive understanding of the ultimate goal of our work. The attacker model aims to provide a more formal version of this intuition, based on the state of the art.

3.1.1 Defining Misbehavior

In literature, there are many different definitions of misbehavior [75, 72, 73, 100, 2], 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. This as opposed to intrusion or attack, which usually implies an attacker attempts to breach a network (e.g., as in intrusion detection, which we discuss below). In addition, 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 misbehaving node is the type of node we should detect. However, the literature often distinguishes [55, 58, 47, 56] 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 failures 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. 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; we provide some examples in Section 3.1.3.

It is important to remark that these definitions are not consistently used throughout the literature. In this work, we use the definitions as stated above.

In the context of fault tolerant distributed systems, these definitions of faulty and malicious nodes roughly correspond to "normal" faults and Byzantine faults. This means that, at least in theory, the concepts of quorums, Byzantine quorums and other results from Byzantine fault tolerance could be transferred to the setting of cITS. In particular, we could transfer the result that at most $\frac{1}{2}$ of nodes must be honest to detect errors, and that $\frac{1}{3}$ of nodes must be honest to ensure the correctness of the algorithms that are executed [12]. However, as we discussed in the introduction, these results rely on a non-ephemeral network setting and requires that the algorithms running on each node are the same. In practice, this will not be the case, because cITS will be deployed in OBUs of different manufacturers and the initial state of each vehicle will not be consistent. In addition, this does not account for different perceptions of reality (e.g., different sensors observing the same phenomenon due to damage, different manufacturing processes or different frame of reference). However, on a higher level where misbehavior reporting is used, game theory can be applied to achieve some similar results, as demonstrated in [57].

Another related term is *intrusion detection*, a popular concept from the field of network security [13, 29, 10]. Similar to misbehavior detection, the goal is to detect messages that are malicious or faulty, and to protect the network from these messages. However, misbehavior detection, most traditional intrusion detection mechanisms are agnostic to the semantics of the payload of these messages. A typical intrusion detection system analyzes the header semantics or content to detect attackers using one of three types of detection: signature-based, specification-based and anomaly-based detection. Signature-based detection recognizes pre-defined patterns, such as specific malicious packets or fragments thereof (e.g., simple pattern matching). Anomaly-based detection employs a variety of techniques, typically from machine learning, to detect packets that deviate from normal behavior. This includes content analysis (e.g., N-gram analysis [91]), but is designed to be agnostic to the semantics of packets in order to provide generic detection for different types of networks. Finally, specification-based detection uses a well-defined specification of a protocol to detect messages that deviate from the protocol. Misbehavior detection could be considered a fourth type of intrusion detection, which uses payload semantics to detect attacks. This type of detection is not feasible for normal networks, because of the large variety of applications; however, in cyber-physical systems (CPS), the physical aspect allows semantics to be used in a meaningful way.

Unlike these approaches, misbehavior detection attempts to detect malicious packets by considering the semantics of the content. There are many good reasons to avoid a focus on content in a generic networking setting, such as a business network: unlike in cITS, these networks see a lot of encrypted traffic and a large variety of application layer protocols. In cITS, we have relatively few protocols, and the transmitted data is typically of a public nature, so the content is not encrypted. In addition, misbehavior in cITS is a greater challenge due to the privacy requirements in these systems. Therefore, misbehavior detection can also include detection mechanisms that study both the raw values and the semantics of message contents. Next, note that a cITS does not have a clear system boundary that can be protected by the intrusion detection system. The goal of a cITS requires a degree of cooperation between the nodes: detecting intrusions into the system is not sufficient. Finally, misbehavior detection includes the detection of faulty nodes, which are not considered by intrusion detection.

3.1.2 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 cITS. Unlike most network scenarios, there is no such universally accepted attacker model that is used consistently for cITS. In this survey, we use the attacker model discussed in this section as a basis for our discussion of misbehavior detection schemes. However, we will occasionally deviate from it to accommodate the specific definitions of some schemes.

The attacker model from [72], one of the seminal works on security in cITS, presents the following four classification dimensions for attackers, and a variety of basic and sophisticated attacks. The distinction between *insider and outsider* concerns whether the attacker possesses certified key material, meaning she is part of the cITS (insider), or not (outsider). This may be extended to include the number of pseudonyms available to the attacker. Secondly, 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 the network or cause harm to other participants. 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 survey, 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 then refers to one or more attackers distributed in a small region, such as a highway section between to 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.

We note that the attackers in this classification mostly correspond with a standard Dolev-Yao attacker model, except for certain insiders. Specifically, the case where legitimate users with certified key material transmit malicious information is not covered by this model. A related concern, also noted by [72], is the potential for Sybil attacks, where an attacker may obtain multiple key-pairs to circumvent security mechanisms. Many authors assume limited capabilities for Sybil attacks, meaning the attacker has possession of a small set (usually 1-3) of valid pseudonyms for each vehicle she controls. This is an issue because for privacy reasons, each vehicle will have possession of multiple keys. We discuss the implications of different approaches to the trade-off between pseudonyms and security in a separate discussion in Section 3.4. Finally we note that like many authors, [72] considers 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, 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. Finally, the authors note that denial of service attacks, where an attacker attacks the limited resources available to the network participants, may not be preventable. Since our goal is detection, we include these attacks, although we agree that we may not be able to mitigate them in any meaningful way.

However, the attacker model consisting of the full Dolev-Yao model, extended with denial of service, malicious insiders and the associated Sybil attacks, is too strong to provide any meaningful level of security. The inherent reason for this relates to the definition of misbehavior and misbehavior detection – attacks are often purely semantic in nature. Thus, 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), she can selectively deliver only those messages that confirm the view presented by the attacker. However, in reality, the attacker cannot arbitrarily arrange delivery 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 significantly enhanced, attackers are considered to have a limited physical presence [63], which also requires that attackers are constrained to physical limitations like the speed of light. Similarly, it is assumed that an attacker cannot arbitrarily re-arrange the traffic without being detected, because this requires that a message is jammed while the attacker receives it. Even if this were possible, the attacker would need to replay the message in a specific way to avoid detection of retransmission by other nodes.

Finally, we remark that it is possible that an attacker can convince a vehicle of a particular situation by exploiting his knowledge of the algorithms and finely tuned jamming. Although this is a direct consequence of the Dolev-Yao model combined with the possession of valid key material by the attacker, it is important to consider this aspect in many detection mechanisms that rely on collaboration between honest nodes. Attacks on those detection mechanisms are explicitly in the scope of this survey, as they provide the most interesting challenge for the design of new mechanisms. However, we do not consider more traditional attacks like viruses, as custom code will not run directly on the OBU, and we cannot prevent misbehavior caused by a sufficiently wide-spread virus. More specifically, we assume that the OBU of a vehicle can be trusted by the detection mechanisms of that vehicle.

In the literature, some authors have attempted to formalize these attacks and the effects on specific mechanisms using game theory [4, 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.

3.1.3 Attacker Types

Because of the wide definition of the attacker model we introduce, many detection mechanisms could be considered unsatisfactory. However, the mechanisms are often secure against specific types of attackers that possess a subset of the capabilities we discussed above. In this section we define attackers that are realistic, based on the literature [72, 7, 63, 53]. We will use these example attackers as a point of reference while discussing the state of the art.

Naive attacker The naive attacker is a casual attacker that performs very simple attacks that do not require special capabilities. This includes damaged sensors or software bugs that cause bogus readings or messages to be generated. Such attacks always affect only the information transmitted by the misbehaving node and typically has a localized scope, as it is directed at the neighborhood of the vehicle. As a practical example, consider a vehicle that transmits that its current speed is 40 m/s on a congested highway, while its previous beacon contained a speed of 4 m/s, then this vehicle can be considered a naive attacker. We note that even very simple attacks by a naive attacker can have a potentially disrupting effect. Consider the inverse of the previous example: a vehicle reports a speed of 0 m/s on a highway, while its the speed in its previous beacon was 40 m/s. Did the vehicle misbehave with its second message, or was the sudden drop in speed caused by an accident? This example precisely illustrates what a misbehavior detection mechanism should be able to determine, given additional contextual information.

Selfish attacker The selfish attacker is comparable to the naive attacker in terms of resources that are at her disposal. However, the selfish attacker is more active and intelligent, with a specific goal: gaining an advantage on the road. This includes controlling traffic lights, simulating traffic jams to obtain an open road, and faking the approach of an emergency vehicle. A selfish attacker exploits all available information and message types that can contribute to an attack, but adheres to message standards to ensure the correct response from the target vehicles. Like the naive attacker, the scope of this attacker is local. A secondary goal of the selfish attacker is to avoid detection, especially when sufficiently high financial or legal penalties are introduced in the system.

Disruptive attacker The goal of a disruptive attacker is to disrupt the cITS, potentially including the associated back-end systems. This disruption can vary greatly, from complex denial of service attacks to simple channel jamming and attempts to cause harm to other participants. Unlike the selfish attacker, the disruptive attacker does not gain an advantage on the road. The disruptive attacker typically disrupts her direct communication neighbors, meaning the scope of this attack is also local. The attack may be directed against the network (jamming, denial of service), or against applications and their users (creating traffic jams or crashes). The disruptive attacker is simultaneously the most versatile and the most dangerous type of attacker. In addition, this attacker may not be discouraged by high fines, so vehicles should be able to deal with these attackers directly. A good example of a relatively simple disruptive attacker we want to detect is an attacker that executes a wormhole attack, which generates false messages in order to disrupt a routing protocol.

Collaborative attacker The collaborative attacker is similar to the selfish attacker, but consists of multiple attackers working together as one attacker. This makes the attacks more difficult to protect against, as collaborative attackers may be able to obtain a local majority to convince other vehicles. We include this example to illustrate a disadvantage of mechanisms that require an honest majority assumption. Some works require only a global honest majority, while others also require a local honest majority. The difference is that a local honest majority requires that at any time, a vehicle must have more honest neighbors than dishonest ones. When a local honest majority is not achieved, the collaborative attacker may be able to convince benign vehicles and use them to further spread malicious data over a larger region, meaning this attacker has the scope *extended*. A collaborative attacker's goals may either align with those of a selfish attackers, seeking an advantage for the collaborating vehicles, or with those of a disruptive attacker. One notable example of collaborative attackers is related to malware spread among vehicles: an attacker can use such tools to force other vehicles to collaborate in the attack.

3.2 Taxonomy of Misbehavior Detection

We have designed a taxonomy to effectively discuss mechanisms to detect attacks within the attacker model discussed above. Our classification is shown in Figure 3. We classify mechanisms based on two aspects, for a total of four types of mechanisms. The first aspect we use for classification distinguishes between node-centric and data-centric mechanisms, which has regularly been used in the literature. The second aspect focuses on the source of the messages that are analyzed – they may either analyze messages from individual vehicles, or from many different vehicles, which we characterize as autonomous and collaborative, respectively.

For our purposes, node-centric mechanisms defined as mechanisms that primarily concerned with the participants of the network. For example, they can verify the behavior of a node by analyzing packet frequencies, correctly formatted messages, and so on, or they are focused on participants of the protocol, verifying their trustworthiness based on the correctness of previous messages. Obviously the correctness needs to be tested some other mechanism; these mechanisms are usually data-centric in nature, meaning they use the content of the message to determine its validity, independent of who transmitted the message. This can be done using models of the physical world, or by testing consistency with messages received from others. Data-centric and node-centric misbehavior detection are mostly orthogonal, in the sense that they can be improved independently. For this reason, many recent authors propose combinations of both types.

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 deduce which messages deviate from the majority. A significant advantage of autonomous detection, based on data from a single source, is that the mechanism will function independent of any attackers that may be present, while 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.

It may seem that autonomous misbehavior detection mechanisms are the only reasonable way to approach the problem, because the attacker model requires that the attacker is allowed to inject his message and then jam all other messages for an indefinite amount of time. However, note that many autonomous mechanisms provide only a rough estimate of correctness, and it is possible for an attacker to pre-compute the results from the autonomous mechanism, so he can produce messages that will always be accepted. This resulting

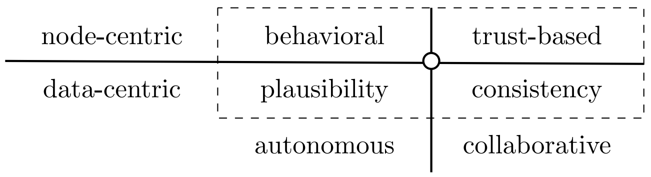


Figure 3: Taxonomy of misbehavior detection.

bound can be exploited by employing collaborative mechanisms to provide a higher quality of detection than is possible using either approach individually. Indeed, the literature is progressing towards these ideas, as evidenced by several detection mechanisms we discuss in this survey.

3.2.1 Node-centric Misbehavior Detection

Node-centric misbehavior detection covers behavioral and trust-based detection. In both of these settings, the primary focus of the mechanism is to establish the trustworthiness of a node. This as opposed to data-centric misbehavior detection, which focuses on the trustworthiness of received information. Behavioral detection has parallels with anomaly- and specification-based intrusion detection, while trust-based detection assigns confidence to a message based on its source(s), often incorporating reputation systems.

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. 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 [59] that was introduced for the security of routing in mobile ad-hoc networks. In a Watchdog, each node monitors the network to verify that its neighbors that are supposed to forward messages actually do so, by listening for retransmissions.

Trust-based The other branch of node-centric detection is trust-based. Trust-based mechanisms exploits 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 3.4. 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 make the step towards revocation simple. However, the ephemeral nature of cITS 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 so-called Sybil attacks [20], where an attacker abuses the reputation mechanism by creating multiple identities. This technique 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.

3.2.2 Data-centric Misbehavior Detection

Data-centric misbehavior detection is focused on using application data from the participants to verify trustworthiness of packets. We classify data-centric detection mechanisms by considering the source of data they analyze, leading to two categories: consistency and plausibility. The underlying idea is that for plausibility, an underlying model that predicts the data exists, while consistency relies on exchange of information between neighbors. In both data-centric scenarios, a major challenge is to deal with inaccurate data, which can be caused not only by faulty or malicious nodes, but also by sensors themselves. In addition, data-centric mechanisms do not necessarily provide the necessary information to perform revocation.

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.

A challenge for consistency-based solutions is that they may only be able to detect the existence of conflicting data, without being able to determine which of these corresponds to reality. This may occur when one group of vehicles reports a traffic jam, while another group denies it, and these groups contain the same amount of vehicles. In general, this type of situation can occur because consistency is checked between multiple senders. Similar to trust-based mechanisms, consistency-based detection requires that an honest majority exists – often even a local honest majority, because consistency mechanisms operate on a shorter time scale. Thus, if several colluding attackers surround a particular receiver, they may be able to exploit consistency to convince the receiver of a particular situation. However, consistency-based detection is still considered to be one of the most promising mechanisms: given an honest majority, consistency can detect attacks even when a node is fully trusted before executing an attack, which is a problem for many trust-based mechanisms. Finally, unlike plausibility-based detection, consistency-based detection does not require a specific, pre-defined model of the data in order to perform its checks, allowing a wider set of data to be analyzed.

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. In addition, plausibility could be used to attempt to distinguish faulty and malicious nodes. 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.

3.3 Scope of Detection

In addition to the taxonomy described above, we can divide mechanisms by where they operate. Figure 4 describes this distinction with three different layers; local, cooperative and 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). 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

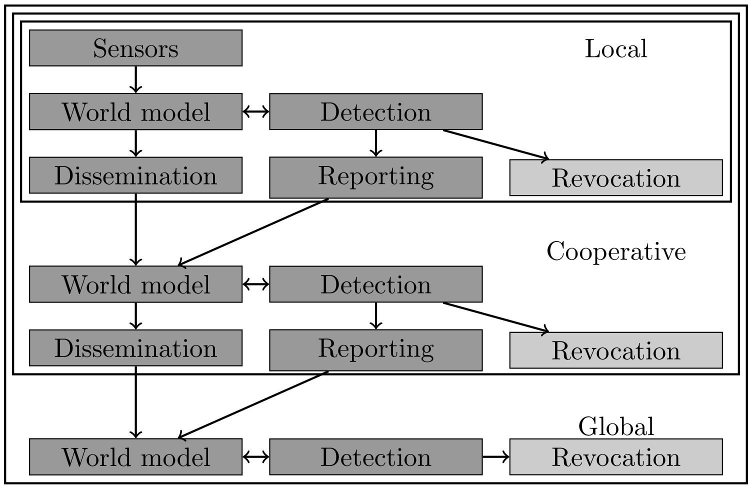


Figure 4: Different scopes at which detection mechanisms can operate.

exist as to what constitutes a cooperative setting, relating to the amount of attacker-controlled vehicles that are in a specific region (refer to Section 3.1.2 for more details).

Most detection mechanisms in the behavioral and plausibility classes operate purely locally, which makes them invariant to some sybil attacks, but 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 operate as a reputation mechanism immediately incorporate a mechanism to perform reporting and/or revocation, both cooperatively and globally, as this is typically a very straight-forward extension of the scheme. Although our primary focus in this paper is detection, and we do not analyze work that focuses on revocation, we remark that this is a very useful property.

3.4 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. However, linkability is an important aspect of many models used for misbehavior detection. Therefore, we will discuss in more detail how pseudonym systems and misbehavior detection interact. The analysis serves to check which detectors will work under a given type of pseudonym scheme and vice-versa, giving an idea of the impact of this choice on the amount of available detection schemes.

Full linkability means that all messages transmitted by the same OBU can be linked. This typically means no changing pseudonyms are used at all. This is the simplest scenario, but it is not feasible to implement from a privacy perspective. Nevertheless all possible misbehavior detection mechanisms could be realized at this level and some types of misbehavior detection even require this level of linkability. Examples of such schemes include trust-based approaches like OREN [5] and [71].

Explicit linkability refers to any type of linkability that allows direct access to an identity. It can be split into three classes, all of which provide technical means for identification of a user. 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 (which we refer to as *trusted third party (TTP) Mapping*) for additional privacy. Using each of these classes, it is technically possible to link pseudonyms, but this capability is somehow limited.

In the first and most complex case, linking will only occur in case of a particular form of misbehavior, using a cryptographic mechanism that is analogous to the revocation of privacy in electronic payment systems. In such a system, when a coin is spent more than once, this misbehavior inherently reveals the identity of a user. This principle has been successfully applied to pay-as-you-drive systems, most notably PriPAYD [90]. However, compared to cITS, such pay-as-you-drive systems have a very limited set of information that can be transmitted, while for cITS, the packets consist mostly of numbers in large ranges (such as speed and GPS coordinates). As the cryptographic mechanisms rely on a small set of valid and known messages, it may not be possible to transfer these ideas to cITS without novel cryptographic mechanisms. However, should such a mechanism exist, it would allow a purely node-centric approach without breaking privacy, avoiding the reliance on data-centric mechanisms that may be affected by sensor noise.

In the case of *back-end mapping*, the back-end can reduce all pseudonyms to the real identities and use those to track vehicles and perform misbehavior detection on that basis. Here, one must rely on the trustworthiness of the back-end system, as it can revoke the privacy of the users. This is especially effective when a direct connection to the back-end is available, either through RSUs or another technology, such as UMTS or LTE. Should such connectivity exist, back-end mapping could enable powerful and effective misbehavior detection schemes in the back-end, at the cost of significant computational effort and privacy from the vehicles towards the back-end. An example of such a powerful scheme is [6].

The last case, *TTP mapping*, requires that more than one entity collaborate to reveal the link between two pseudonyms, which is hidden using a protocol like VToken [81]. This is also called legal or organizational linkability, because it allows legal or organizational processes to be applied to linking messages, as opposed to a purely technical solution. A direct implication is that linkability will incur a significant delay and can only be performed in the back-end, as the legal system requires time to authorize it. This time is used to perform the necessary trade-off between privacy and security, which is the preferred situation for most privacy advocates. This provides some additional challenges for misbehavior detection: It is now only possible to link two pseudonyms when it has been shown that this is necessary. Although "necessary" is very vague, what we can expect based on existing laws surrounding topics like search warrants is that there must be a suspicion of a crime, which provides strict legal requirements. We note that a TTP mapping will not allow misbehavior detectors in the back-end to function, unless their dependence on the identities associated with messages is limited, or the necessary information is provided by other vehicles. However, we can use misbehavior detectors to verify claimed misbehavior, which facilitates the revocation process, where the information available to the detectors is comparable to that of the full linkability scenario. Additionally before being able to verify claimed misbehavior, the misbehavior must first be identified through other means: we thus require that powerful detection schemes be employed at the local and cooperative levels, including the detection or mitigation of Sybil attacks. As discussed in Section 2.4, variations of the organizational mapping are being implemented in the upcoming generation of standards.

All this different classes of explicit linkability require that the back-end is able to observe messages through RSUs or, indirectly, using misbehavior reports from vehicles. This means that detection will likely not occur everywhere, or incur significant delay, because a permanent and direct connection to the back-end is not assumed.

Implicit linkability or *inference* allow 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, [35] and [93] are good examples, while for inference on message content, a good example is the Kalman filter [87, 86]. 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 cITS 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, the authors remark that even in this scenario, tracking is possible simply by following a vehicle. This is a more appropriate baseline to test privacy against, and a requirement of *no linkability* between messages is far stronger.

As the no linkability and full linkability are theoretical scenarios, we are left with a choice between two types of linkability, explicit and implicit linkability, which are illustrated in Figure 5. These types of linkability can be implemented in different ways, which are illustrated as white boxes in the figure; depending on the implementation, linkability may be performed by different entities. For example, signal correlation (two messages from the same source) is typically only possible locally, if at all. On the other hand, if the certificate for a pseudonym contains information about the vehicle, as foreseen in current standards, anyone can perform linking of the appropriately signed messages. If appropriate cryptographic mechanisms are used, such as group signatures, then only entities in the back-end or a TTP can resolve this random value to the identity of the vehicle. Note that we have classified the different options with respect to the layers of detection from Section 3.3.

4 State of the Art

In this section, we present state of the art mechanisms for misbehavior detection in cITS, as structured by our taxonomy. To provide insight in how this state of the art developed, we also include important related mechanisms designed for mobile ad-hoc networks (MANETs) and wireless sensor networks (WSNs). We note these discussions are not intended to provide a complete overview of the state of the art in those fields – separate surveys exist for them [19, 99]. After discussing the individual mechanisms for our state of the art, we provide a summarizing table of our analysis in Section 4.3. This summary will include the additional parameters we discussed in Section 3; the scope of the detection and its compatibility with pseudonyms.

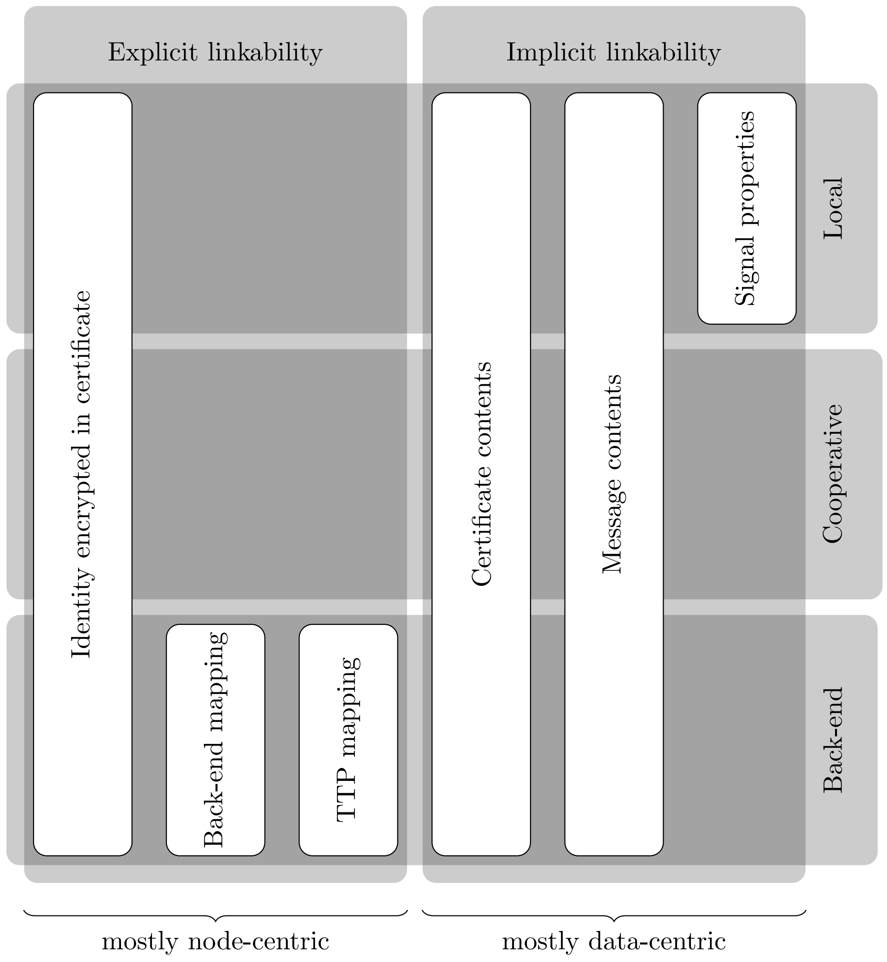


Figure 5: Relationship between the level of detection and the distinct linkability mechanisms.

4.1 Node-centric Mechanisms

The idea of node-centric detection mechanisms is to use knowledge about the senders of messages in order to detect malicious senders. Signatures are typically used to achieve this goal, with certificates provided by a PKI (cf. Section 2), that allow identification of a messages' sender. Once the authenticity of a message sender is verified, it can be used to correlate messages originating from the same sender and thereupon analyze its behavior. Such analysis can include whether the message frequency, headers and content are in line with protocol specifications. Using past behavior, we can then assign a trust value to information sources based on previous interaction with them. The assumption is that malicious messages most frequently originate from information sources that have misbehaved in the past.

In our taxonomy, we classify node-centric mechanisms as either behavioral or trust-based. In earlier work for MANETs, node-centric mechanisms are the most popular, because these networks are focused on providing an abstract service to network participants, as opposed to a set of services that is inherently tied with data semantics, as in cITS.

4.1.1 Behavioral mechanisms

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 cITS typically focus on identifying nodes which send messages too frequently or nodes which modify the message content in 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 cITS-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 cITS scenarios.

Before discussing mechanisms specifically designed for cITS, we take a brief historical perspective and discuss a seminal work developed for MANETs [59]. This paper introduces two tools for misbehavior detection: the Watchdog and the Pathrater, which they evaluate for the multi-hop routing mechanism called dynamic source routing (DSR). The essence of the watchdog mechanism is that each node that participates in routing monitors the network after forwarding a packet to a next hop. This node can then overhear whether the next hop forwards the packet or not, and therefore establish whether it is correctly behaving as defined by the protocol (in this case, DSR). Because a lossy channel might cause transmissions to be lost, a Watchdog should be configured with a threshold before it detects a node as malicious. Challenges for this mechanism include loss or collision on the channel, as well as false reports generated by malicious or colluding nodes. The aspect of colluding nodes is discussed in more detail in Section 4.1.2, where we discuss trust-based mechanisms that deal with this issue. The second tool from [59] is Pathrater, which uses the watchdog results to rate the different network paths, so that the routing mechanism can select the most trustworthy path even under the influence of attackers.

Routing-oriented Mechanisms to detect attacks on routing have the attractive advantage that they are fully independent of content. For this reason, they are common in MANETs, and some authors address the challenge of translating these results to cITS' more specific requirements.

Hortelano et al. [40] have evaluated the usefulness of the traditional Watchdog mechanism for cITS. In their Watchdog, each vehicle observes the forwarding behavior of its neighbors. Because the routing mechanisms used in the network are known to each vehicle, it can predict which packets should be re-broadcast by neighboring vehicles. Hence, the ratio of packets that *should* be forwarded by neighboring vehicles versus the number of packets that are *actually* forwarded can be used to measure the protocol adherence of these 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*). Overall, the presented approach is independent of message content and can be adopted for different routing mechanisms. 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 set the threshold for malicious behavior detection, which is a globally fixed parameter in their scheme. This is a traditional trade-off between false negatives and false positives. Hence, dynamic adaptation of thresholds for watchdog mechanisms would be necessary. In addition, we note this paper does not address privacy in their analysis, making the true suitability for cITS unclear at best. Last, the authors note that their system is vulnerable to several attacks. Essentially, these are the same as those that apply to the MANET Watchdog mechanisms; in addition, there is no protection against Sybil attacks. Both of these issues are considered as future work. We note that in our more general attacker model, the attacker could use selective jamming in addition to further amplify several attacks.

Jamming detection In addition to mechanisms that are designed to detect attacks on routing, a very useful mechanism involves the detection of another type of attack: jamming. The assumption is that a jamming attacker behaves intelligently, in order to gain an advantage through his jamming, instead of simply performing a denial of service attack.

The previously discussed behavioral mechanisms focus mainly on detection of attacks on the routing layer, that is, on message dropping, alteration, and replay attacks, under the assumption that the attacker can somehow execute that attack. In contrast, Hamieh et al. [38] describe a detection mechanism for jamming attacks based on detecting patterns in radio interference, which is one of the most feasible approaches to implement those attacks. The assumption is that an attacker will intelligently jam the radio signal only during the time where honest vehicles transmit. This approach, known as selective jamming, is a common technique to avoid being easily detected due to constant jamming of the wireless channel. The proposed approach makes use of the fact that a selective jamming attacker will wait until regular transmissions occur until she jams the wireless medium. Hence, a correlation coefficient between correct reception time and time where errors occur is calculated. If the correlation is high, that is, if the medium is jammed most of the time when regular reception should occur, the medium is considered jammed. In order to achieve useful results, the authors took into account realistic reception and error probabilities as a baseline. Only if the correlation is unusually high, the medium can be considered jammed. The proposed method is interesting because the correlation can be passively calculated with a simple formula, and because detection selective jamming is an important problem that is often neglected in security-related works. A deeper discussion and classification of jamming attacks is provided in a later work by Puñal et al. [69], which discusses different types of jamming attacks and analyzes their feasibility. In particular, they show some types of jammers are capable enough to prevent communication altogether with high probability, illustrating the necessity of jamming detection.

4.1.2 Trust-based mechanisms

Similar to behavioral mechanisms, many trust-based mechanisms for cITS are rooted in mechanisms that were developed for MANETs. These partially evolved from mechanisms such as the Watchdog [59] (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 [59] 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 cITS, as the connectivity between vehicles is sporadic, and privacy requirements lead to a vehicle being allowed to use multiple identities.

Trust-based mechanisms allow the participating nodes to vote on the correctness of data, or the trustworthiness of other nodes. Therefore most of these schemes employ some method of voting or agreement among nodes, typically relying on an honest majority. In the past, a number of common schemes have been surveyed in [2], which will be taken into account in the following. The main focus are safety applications where vehicles broadcast messages to warn other vehicles about events like dangerous road conditions or accidents. Problems arise when misbehaving vehicles claim false events, and, hence, vehicles receive conflicting information about specific parts of the road, as shown in Figure 6.

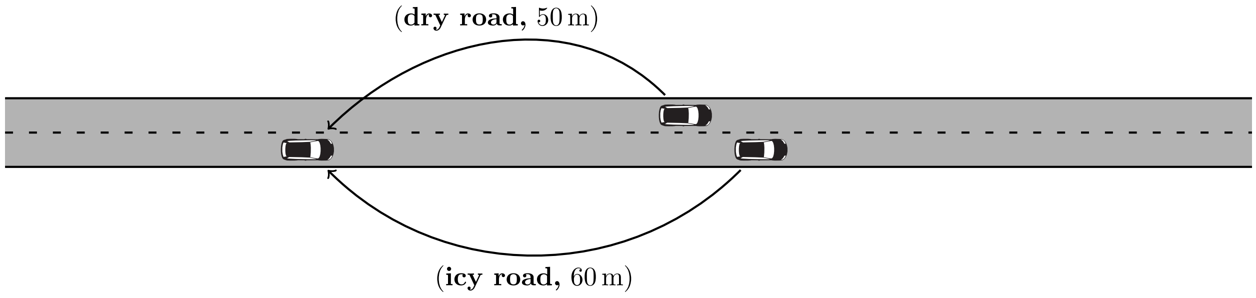


Figure 6: Example showing conflicting event notifications.

Voting-based event validation Although typical trust-based mechanisms express trust in nodes, this trust can be difficult to transfer over many hops. Voting-based event validation addresses exactly this challenge: how can the validity of an event be transferred efficiently? This is done by allowing vehicles to cryptographically vote, essentially vouching for its validity.

The work by Cao et al. [11] determine the correctness of event reports through voting. The core contribution of the paper is an efficient way to collect signatures from a sufficient number of witnesses without causing too much bandwidth overhead on the wireless channel. In the proposed scheme, growth codes are used to achieve an efficient dissemination of the witnesses' signatures on the claimed events. Growth codes are a type of erasure code, and are used to efficiently XOR together signatures over an unreliable channel for efficient collection. They are optimized to decode as many signatures as quickly (in terms of transmission rounds) as possible; each of these signatures certifies a claimed event, and only events with many valid signatures are accepted as valid. Growth codes give a relatively strong confidence, but introduce a significant delay before convincing a receiver of the validity of the message, because a threshold amount of signatures must be decoded first. A strong criticism of this approach is that false negatives may cause significant problems; if insufficient signatures are received, events may be missed completely. In addition, the attackers are assumed to have exactly one key pair each. However, this applies only on a per-event basis (meaning the scheme is fundamentally compatible with pseudonyms, given that any user is bounded to exactly one valid pseudonym at any time).

Another similar approach has been presented by Hsiao et al. [41]: here, the main goal was to determine whether a claimed event has actually happened. In order to prevent attacks, the senders collect a number of witnesses for each possible event. For space efficiency purposes, z-smallest probabilistic counting is used, reducing the required amount of signatures that need to be attached to the message. The idea of z-smallest is that, given n elements uniformly distributed between 0 and 1, the z-smallest element gives an approximation of n by calculating $\frac{z}{c}$, where c is the value of the z-smallest element. To protect against inflation, that is, attackers that try to increase the number of witnesses of an event, each vehicle signs a hash of its vehicle id, the event type, location segment, and time of the event. Only the z-smallest signatures are kept with the aggregate. The attacker cannot produce enough signatures on hashes that fall into the z-smallest values, because at least z hashes can be verified by the receivers. Therefore, an attacker cannot artificially increase the result. Important to mention is that there is no deflation protection in this scheme; the attacker can reduce the amount of signatures attached to the message. The authors argued that an attacker will only try to produce fake events, such as a fake accident, and not try to hide events. Hiding events may also be achieved more easily by a powerful jamming signal. Similar to the previous paper, it is a hard requirement that vehicles are restricted to exactly one pseudonym per time period as the attacker model does not include Sybil attacks.

Decision logic More traditional approaches to trust can be applied by using various types of decision logic to decide on the trustworthiness of vehicles based on previous behavior. However, these schemes are difficult to employ correctly in highly ephemeral networks such as cITS, which is the main challenge addressed by work in this category. The attacker model for these mechanisms necessarily excludes Sybil attacks – detection mechanisms that identify multiple identities as being the same vehicle are necessary to make these schemes effective.

One interesting concept has been presented by Raya et al. [74] in 2008. They proposed a mechanism to judge whether incoming messages are trustworthy by analyzing incoming traffic using different factors. The authors note their scheme is data-centric, because they use their mechanisms to evaluate confidence in messages. However, we concentrate on the *trust establishment* proposal of their work, which is a node-centric approach; they analyze how data-centric results can be used to establish trust between nodes. The authors used a combination of three different factors: 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 output and input to the decision logic, which then decides whether or not to trust a given piece of information, by evaluating a trust in the sender. 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 [85, 17] was identified as the most promising technique.

Default trustworthiness	bases trust on certified attributes of vehicles (e.g., police cars).
Event- or task-specific trustworthiness	bases trust on the observation that certain types of vehicles are more likely to report about certain events.
Dynamic trustworthiness	captures time-variant functions, such as the closeness of a reporting vehicle to the reported event.

Table 1: Trustworthiness factors used in [74].

Similarly, Rawat et al. [71] discuss the usage of Bayesian logic to predict the trustworthiness of vehicles. Basically, they compute the maliciousness probability for a vehicle i for a time t given some observation O_t. Their approach relies on basic Bayesian reasoning, i.e., computing $P(T_i\text{malicious}|O_t)$ by applying Bayes' theorem. This is actually quite similar to the data-centric idea proposed by Golle et al. [31], except that here the maliciousness of vehicles is modelled instead of the correctness of the data. Unfortunately the authors do not specify how they obtain the necessary conditional probabilities (such as the probability of a message given previous observations). The scheme seems to require a pre-configured knowledge of the probabilities of how likely it is that a particular message is received. Another issue is that the authors' model does not *forget*: they always compute probabilities over the entire observation history. They do point out their detection threshold will change over time, but do not specify how. In their evaluation, they contrast signal-to-noise ratio with their computed trust values, based on a model of signal-to-noise ratio over relative distance and receive power. This makes computation of the associated probabilities easier: if the claimed relative distance is falsified, their scheme can reliably detect this based on their model. However, this approach may be difficult to generalize. Also, their results only work because the model assumes a fixed transmission range, which is also used by the attacker. Note, however, that a non-standard transmit power could be detected by other nodes if this approach is applied and the trust values are shared.

Consensus mechanisms Consensus mechanisms are node-centric approaches that use assigned trust in addition to some data-centric inputs to compute a consensus among several nodes to prove the trustworthiness of vehicles.

Following several papers focussing on data-centric misbehavior detection, discussed in Section 4.2, Leinmüller et al. also designed a cooperative trust mechanism [52] to improve the results of some of their mechanisms by exploiting trust developed between vehicles. The goal of the mechanism is to defend against attacks performed by static roadside attackers. This is done by combining the minimum distance moved (one of the data-centric mechanisms) and transitive trust. First, each vehicle evaluates locally whether sensor information implies that the sender of a message has moved a certain minimum distance. Then, each vehicle broadcasts the collected information, indicating which neighbors passed the local the minimum distance moved check. The assumption is that if a vehicle A receives positive feedback about a vehicle C from vehicle B and A has earlier identified B as trustworthy, then A will also trust C without directly verifying it. The scheme also incorporates neighbor information, which is shared among vehicles using periodic messages (proactive), although the authors also note that reactive mechanisms may be possible. An honest majority assumption allows this mechanism to work, but only when the attacker is (mostly) stationary, which is a fairly weak attacker model. Nevertheless, the work provides an example of how a simple consensus mechanism can be used to increase security.

Hao et al. [39] developed an RSU-backed, group-signature-based pseudonym protocol, called cooperative message authentication protocol (CMAP), which also aims to exploit consensus. This mechanism is mainly intended to reduce computational effort necessary for authentication in cITS, by exploiting an overwhelming majority of honest participants. The idea of the protocol is to have a small set of verifiers in the transmission range of a sender verify the message, which broadcast a cooperative authentication message to indicate the message was correctly signed. Given sufficient of these messages, the original message is accepted as legitimate. This reduces the overall required computational effort required to verify messages; instead of providing immediate misbehavior detection, this work builds a consensus protocol to replace existing proactive security with a more efficient variant. The keys used to authenticate messages are generated by RSUs, and the scheme thus requires complete RSU coverage at all road entrances and exits. A minimum amount of network coverage is also required, in order to achieve the minimum amount of verifiers with high probability. The authors claim they can protect against malicious or compromised RSUs, and that the scheme could be improved by a centralized data-centric misbehavior detection mechanism in case of conflicts. Unfortunately, collusion attacks are not covered in the paper, exposing the scheme to a significant security vulnerability unless further data-centric mechanisms are employed. Finally, in terms of privacy, the scheme only provides a solution against vehicles, which cannot determine the identity or pseudonym associated with any message. However, because authorities distribute the group private keys to each vehicle through RSUs, they are capable of tracking each participant using a tracing key.

The authors in [65] have proposed a scheme to build consensus about certain events in vehicular networks using a dynamic threshold for their consensus mechanism. This takes ideas from voting-based event validation mechanisms, which focus on events like traffic jams, to a more safety-relevant setting with tighter time constraints. The proposed scheme collects a number of reports about the same event from surrounding vehicles until a certain threshold of supporting reports is passed, after which the message is considered to be trustworthy. To cater for safety-relevant applications, the authors took into account a maximum waiting time, before which a decision must be reached. For instance, in case of an accident warning message, the decision whether to trust the warning must be made early enough, so that the vehicle can still brake in time to avoid further collisions. A main contribution of the authors is to implement the acceptance threshold in a dynamic way. The threshold is adapted based on factors like distance to the claimed event and number of vehicles currently in the neighborhood. However, like most consensus mechanisms, this approach suffers from potential Sybil attacks. The dynamic threshold allows the mechanism to deal with density variations on the road, but as the authors also remark, selecting an appropriate initial value is still an open challenge.

Kim et al. [48] combine several data-centric and node-centric parameters of messages into a single curve, called the certainty-of-event (CoE) curve. The validity of messages is determined by checking messages against the vehicles' own sensors, messages from other vehicles, and, where available, validation by infrastructure. The authors propose that the scheme will only warn the driver if the aggregated outcome of the mechanisms indicates a relevant event occurred. Relevant events are those *valid* events in which the vehicle is interested, that is, the vehicle will encounter the event in a reasonable amount of time. Validity of the message is defined using a so-called a threshold curve, which determines the amount of supporting evidence required before the driver is notified about the event. This allows for fast responses in emergencies with less certainty, while it reduces the false positives for events where more evidence can be gathered over time.

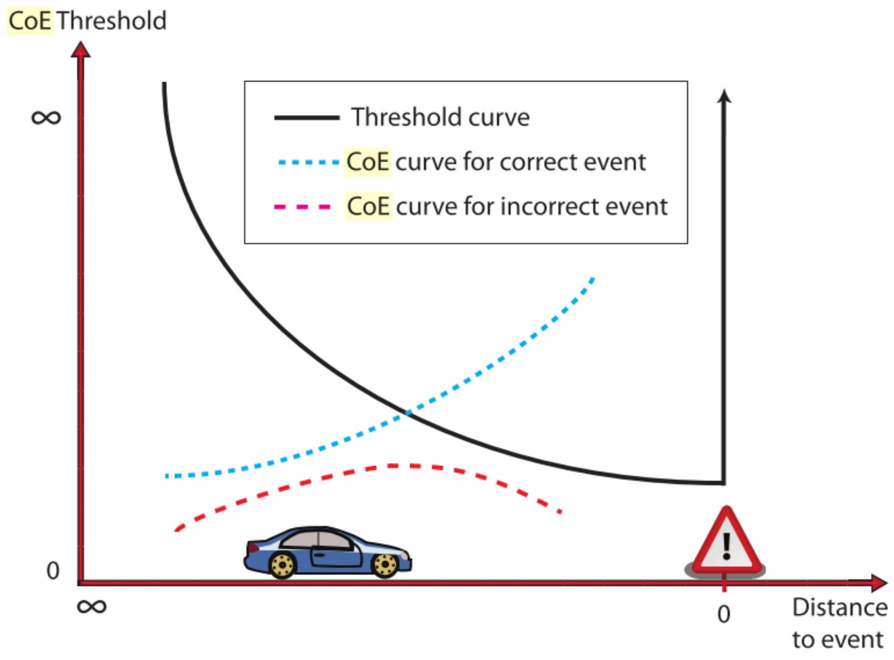


Figure 7: Figure from [48] that shows the intended effect of the CoE curves.

The authors produce an aggregated outcome of their schemes as a CoE curve over time, as illustrated in Figure 7. When this curve exceeds the threshold curve, the driver is notified about the event. We note that the CoE could be vulnerable to Sybil attacks, depending on the implementation of the various sources of information that are specified. In addition, this scheme is very difficult to generalize to other types of applications, because specific locations are required for events. The example application the authors use, the emergency electronic break light (EEBL) application, provides such locations, but it is unclear whether the scheme can deal with multiple lanes and urban settings, where there may be some uncertainty about the driver's path. However, the CoE is a powerful concept that can be employed to allow for maximum certainty before warning the driver, reducing false positive and negative rates.

Voting 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 voting-based event validation in that they allow vehicles to vote on trustworthiness of other vehicles, rather than evaluate the trustworthiness of particular claimed events. As with consensus mechanisms, voting mechanisms strictly require protection against Sybil attacks, and those attacks are thus not part of the attacker model.

Besides their trust evaluation based on the Dempster-Shafer interference [74], Raya et al. [73] 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 mechanism called LEAVE. In their scheme vehicles collect accusations about a likely attacker until the number of reports passes a certain threshold. If enough accusations are collected, the accused vehicle is evicted temporarily. Once a vehicle receives sufficient accusations, it can disseminate an aggregated message that contains the accusation, as well as a sufficiently high number of supporting signatures from other nodes. Vehicles receiving such an aggregated report can then directly ignore the accused vehicle. A core advantage of this approach compared to reputation systems is that the latency of the detection mechanism is much lower; reputation systems require time in order to build trust.

Raya et al. [73] thus 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. A disadvantage 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.

Like the previously discussed approach, the goal of Zhou et al. [100] is to remove misbehaving insider attackers from the local neighborhood or the network. To achieve this goal, two specific mechanisms are proposed: suicide-based local eviction (SLEP) and permanent revocation (PRP). As their names suggest, SLEP evicts attackers locally and PRP achieves global revocation of misbehaving vehicles. Both protocols can be applied even if pseudonym-schemes are used in order to preserve user privacy. The local eviction protocol works using a so-called suicide mechanism. The assumption is that vehicles in the direct neighborhood of an attacker are able to detect bogus messages, for instance, by comparing them to local sensor information about the same event. Once an attack is detected, the detecting vehicle broadcasts a message accusing the potential attacker vehicle. Surrounding vehicles receiving the message will henceforth ignore messages from the accused vehicle, but they will also ignore further messages from the *accusing* vehicle. The price for the accusing vehicle is necessary to avoid fake accusations. To prevent multiple honest vehicles from committing suicide in response to the same misbehavior, a random back-off timer is employed before disseminating accusations. To achieve global revocation, each vehicle periodically reports its local blacklist of possible attackers to the back-end. There, all lists are merged and analyzed taking into account trust levels of the accusers, which are based on their previous reports. Once enough reports about misbehavior are collected for a specific vehicle, its credentials are revoked by the central authority.

In a subsequent line of work, Raya et al. [76] and Bilogrevic et al. [4, 5] have presented closely related game-theoretic approaches to combine their eviction scheme from [73] with a suicide mechanism. The proposed mechanism combines the two previously presented approaches. Once a vehicle detects a misbehaving node, it can choose from three options. Namely, it can

- **abstain** from taking any action,
- **vote** against the attacker by disseminating an accusation, or
- **commit suicide,** which results in the immediate eviction of both the attacker and the accusing node.

Each course of action 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.

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, she 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 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 herself, but by the manufacturer. The authors note that manufacturers may optimize their own cost, which could invalidate the game-theoretic results that were obtained by assuming that all vehicles optimize their individual costs.

Pseudonym linking To improve results from node-based detection mechanisms, vehicles should locally be able to resolve pseudonyms to identify vehicles uniquely. This allows vehicles to employ trust-based mechanisms more effectively, because trust can be built over a longer time, and reduces the risk of Sybil attacks. Many pseudonym linking mechanisms exist among data-centric detection mechanisms, but there are also some cryptographic mechanisms, which we discuss here. Remark that these mechanisms can, by definition, only work because privacy mechanisms for cITS are not perfectly privacy-preserving (i.e., allowing for some linkability) – improvements to those mechanisms may reduce the effectiveness of these detection mechanisms.

A general problem in any scheme that uses consensus building or voting schemes to build trust is that attackers can abuse pseudonyms to mount Sybil attacks. [97] has provided a mechanism to detect such Sybil attacks given a list of signed messages about the same event. Detection is performed by roadside units in cooperation with a centralized trusted third party. To link pseudonyms, the scheme employs a combination of coarse-grained hash values and fine-grained hash values. The coarse-grained hash values are generated with a secret key known to roadside units. All hash values of one vehicle's pseudonyms map to the same coarse-grained hash, but also multiple vehicles' pseudonyms can map to the same hash. Once the roadside unit overhears a number of messages about the same event, it can check the coarse-grained hash values of the pseudonyms used. In case the same hash occurs multiple times, it is likely that a Sybil attack was performed. However, false positives can occur due to collisions of hashes from multiple vehicles. Hence, the roadside unit will forward all concerned pseudonyms and messages to the centralized trusted third party in case of suspected attacks. The trusted third party is in possession of a secret key to check fine-grained hashes from the pseudonyms. These fine-grained hashes will link exactly the pseudonyms of one vehicles. Thus, the trusted third party is able to make the final decision about possible Sybil attacks. A disadvantage of the proposed system is that it is possible for a centralized authority to link all pseudonyms belonging to one vehicle, which severely weakens privacy protection.

Finally, some proposals attempt to work purely within trusted hardware. For example, the authors in [96] present an approach called MisDis, which uses tamper-proof hardware to log every single message sent or received by a vehicle in a secure log. Rather than verifying the messages after the fact, they apply a commitment scheme to require a vehicle to send an authenticator with each message. This authenticator is essentially a commitment on an earlier log entry. Thus, if a vehicle transmits malicious messages (i.e., has malicious entries in its' log), receivers have evidence that a log was falsified. Logs entries can be fetched by time-stamp in an interactive protocol, and then replayed to determine the correct behavior of the vehicle. Key here is that the behavior is deterministic and can be modelled as a state machine, i.e., this step is easy and all implementations are exactly the same. Deviation is then considered misbehavior, which is reported in a similar way to a central authority. This approach has several problems: first, it requires that a single central authority can access the logs of all vehicles remotely (possibly with a time delay). Next, because all transmitted and received messages are stored in the log, including the signed acknowledgments the authors propose, the log will be very large, especially when considering the lifetime of the vehicle. To check a vehicle for misbehavior, the log is requested over the air by a verifying vehicle. The log is encrypted with a session key. However, the biggest weakness in this scheme is the requirement of determinism; the MAC layer in the form of IEEE 802.11p is not deterministic and best-effort – reception of messages is not guaranteed. Closely related to this is that sensor values, such as GPS data or speed data, which are noisy, and therefore non-deterministic. The authors do not discuss how such issues are addressed.

RSU-supported Sybil attack detection There also exist many schemes that operate using a centralized authority to detect Sybil nodes.

An example of such work is [98], which proposes a baseline to verify all messages; to this end, the authors define an attack as follows: given a triplet of time, location and event type, if this triplet is signed by the same vehicle using two distinct pseudonyms, this constitutes an attack. In order to prevent these attacks, the authors propose an inherent linking between pseudonyms based on hash functions. To limit the amount of possible triplets, time- and location intervals, as well as event types, are predefined. To avoid contacting the central authority continuously, the authors delegate detection to semi-trusted RSUs, which may detect suspicious behavior and report it to the central authority. To this end, the generated pseudonyms are divided using coarse-grained groups and fine-grained groups. The coarse-grained groups contain pseudonyms hashed with a key known to the RSUs, while fine-grained groups are those sets of pseudonyms assigned to a single vehicle. Thus RSUs cannot directly identify vehicles, but they can detect that two pseudonyms are in the same coarse-grained group; these cases require the RSU to contact the central authority for further verification. Finally, the authors propose a number of potential improvements to their scheme. We note that the scheme provides privacy against other vehicles and compromised RSUs, but the central authority is trivially capable of obtaining all the information about all the vehicles (given sufficient bandwidth to the RSUs). However, for this sacrifice in privacy, a significant improvement in security can be achieved by defeating Sybil attacks in the presence of RSUs. Finally we note that the scheme explicitly and strongly relies on a single central authority, which may provide implementation challenges, as the amount of vehicles, types of events, time and space granularity are all required to be predefined. This especially makes it challenging to obtain interoperability between national borders.

[15] points out that Sybil nodes that originate from the same vehicle will always have unrealistically similar trajectories over time. Furthermore, they assume that RSUs are available in some places and are fully trustworthy. RSUs send out signatures (presumably on a time stamp), which vehicles can record. To perform Sybil attack detection, a vehicle performs four steps: first, it requests the last r signatures from each neighboring vehicle. Then, it computes the differences between all the trajectories, which can be derived from these r signatures (because the positions of the RSUs are known or in the messages). The vehicle then uses a statistical mechanism to judge whether Sybil attacks exist or not. Finally, the vehicle classifies all the paths in order to consider the different Sybils of a vehicle as one sender, effectively removing the attack. However, numerous problems exist with this approach. First, the position samples are based on the locations of RSUs, which for paths with low RSUs coverage makes it a weak metric. Second, while the position samples are signed and thus cannot be modified, the vehicle is also allowed to set the value *None* for time slots in which no signature was received from an RSU. Another issue is that the ability for vehicles to request sets of position traces provides a serious threat for denial of service attacks, as the bandwidth overhead for each request is significant; for every vehicle that decides to do detection, r signatures per neighbor must be transmitted; for good detection quality, r should be sufficiently large. Another issue is that these requests require participants to give up their privacy in order to prove that they are not Sybil attackers.

Footprint [14] improves on this idea: it is another example of a scheme that exploits central authorities by using similarity of trajectories, which are generated using signed messages by RSUs that a vehicle passes. In Footprint, these trajectories are cryptographically protected, and consist of special signatures, requested by the vehicle from the RSUs is has seen while driving. Footprint works by bounding the potential set of valid distinct trajectories an attacker can create. This bound is, in the worst case, the power set of trajectories, but can be limited in size using a test (which we do not discuss in detail here). The authors assume that real trajectories are sufficiently distinct; by forcing the attacker to obtain signatures through the RSUs, the bound is created based on the real path of the attacker. Then, when detecting Sybil attacks, all trajectories that are suspiciously similar are considered as coming from the same vehicle (referred to as a Sybil community). The authors use the trajectories for every message as an authentication mechanism, which allows any vehicle to compute the Sybil communities and avoid Sybil attacks. The signatures of RSUs are time-dependent and unpredictable, which means that location privacy is achieved against long term tracking.

4.2 Data-centric Mechanisms

In this section, we consider the different forms of data-centric misbehavior detection that were identified in Section 3.2. The core distinction between consistency and plausibility is that consistency analyzes messages from different senders, while plausibility attempts to verify one or more messages from the same sender. Plausibility mechanisms include analyzing physical layer signals to detect that they indeed come from the same source.

4.2.1 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 position verification 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.

[93] provides one of the earliest position verification mechanisms, by verifying claimed positions using signal strength metrics. A problem they identify is how to ensure that there are no Sybil attacks on-going that manipulate this approach; to this end they provide a scheme called improved position verification to exclude the Sybils of a Sybil attacker. The basic scheme consists of three roles: claimer, where a node claims a position in a beacon, witness, where a node receives a beacon and measures its proximity using the received signal strength, which is transmits in subsequent beacons, and verifier, which collects signal strength measurements to estimate and verify the position of a node. In improved position verification, the authors attempt to detect Sybils that are providing witness messages. To this end, RSUs are used, which issue a signature that declares the presence of the vehicle at a particular RSU at a specific time, driving in a specific direction. They then require witnesses to be approaching vehicles (on the other side of the road). The scheme does not account for scenarios with low amount of witnesses, or one-way situations, but their statistical approach that compares RSSI-based estimations with claimed positions is mathematically sound. For sufficient estimates, the scheme will provide a satisfactory solution. After classifying a neighbor as a Sybil node, the vehicle will go through its neighbor table and verify other neighbors in order to find other Sybils originating from the same malicious entity. However, this assumes a targeted attack – a Sybil attack from the other side of the road, simply to create confusion or to accuse a node, may still be possible.

The authors of [35] provide a detailed analysis of Sybil attack detection through analysis of physical layer properties. They assume that antennas, gains and transmission powers are fixed and known to all users of the cITS. However, they allow attackers to modify their transmission power. By applying signal models, they use the received signal strength to determine the approximate distance to the sender and apply this to verify the GPS position transmitted in each beacon message. They show the theoretically possible areas where an attacker can transmit to cause the receiver to observe the desired received signal strength, in order to correspond to the received signal strength. The authors also analyze the effect of using different antenna models (bi-directional and omni-directional) for the receiver. As the authors point out, they do not consider special propagation models or GPS errors.

Similarly, [79] exploits the relation between location, time and transmission duration. They forbid forwarding of old events and discard messages whose positions are not either increasing or decreasing across the road (i.e., the forwarder must always be the middle vehicle between an event and the receiver). The authors explicitly exclude voting based on multiple received events. Before forwarding an alert, the receiver should wait for a fixed amount of time to listen for beacons, which are used to check if an event is actually occurring. Time of flight is used to verify the positions transmitted by each sender, computing the time spent in the air by the message. This is done by checking the following equation: $t_2 = t_1 * \frac{dist(l_{i,t_2}, l_{j,t_1})}{c}$, where t_1 is the claimed time of transmission, t_{j,t_1} is the claimed location of the sending vehicle j at this time and l_{i,t_2} is the location of the receiving vehicle i at the time of reception t_2. However, this does not include delays incurred while reading the GPS data, processing this information and other received packets, nor transmitting the packet to the physical layer. Nevertheless, when considered with a threshold this can work, if the GPS error is sufficiently low and the clocks are fully synchronized and sufficiently accurate. However, it requires that the attacker cannot estimate locations and time with respect to her target that are sufficiently accurate to be considered as legitimate.

Multi-check verification Another early idea was to use several sources of information to verify the general validity of messages, rather than just using signals to verify a position, which was sketched in [50].

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-check 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.

Lo & Tsai [58] have introduced a particular attack called the illusion attack, where the attacker injects false information into the cITS. To protect against this attack, the authors propose a plausibility validation network (PVN), which consists mainly of a checking module and a rule database, which allows information in new messages or provided by ones' own sensors to be verified. The rule database contains a set of rules that govern whether certain information should be considered valid or not, by analyzing the individual fields and verifying them against each other based on the rules provided by the rule database. This set of rules is dependent on message type. A message is valid if it passes all relevant verifications. The authors go on to provide a list of these rules in order to detect fake vehicles, which includes dropping of duplicate messages, that the location should be in range and plausible, the time stamp should be checked and the velocity should be plausible. The authors provide a formula for verification for each rule. When a message is considered valid, no rule has detected an attack, which means that either the appropriate rule does not exist yet, or the message is legitimate. As the authors only consider attackers manipulating sensors (i.e., the attacker does not have key material), this will be capable of detecting most attacks. However, our attacker model allows the attacker to generate arbitrary signed messages, which means that the attacker can generate messages to pass the (known) rule database.

In [54], the authors do not only discuss data-centric consistency mechanisms, but also discuss some plausibility checks, referred to by the authors as autonomous verification, although this does not preclude information from multiple sources. These checks were previously studied in [51] and are combined by the authors with their consistency approach. Here we discuss the plausibility mechanisms in detail. The introduced checks are called acceptance range threshold (ART), mobility grade threshold (MGT) and maximum density threshold (MDT). 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 and is more generally applicable to MANETs. MDT examines the maximum amount of vehicles in a particular area; when too many beacons are sent from one location, the vehicle ignores further beacons from the same area, in order to limit the impact of Sybil attacks. In addition to these checks, the authors mention the possibility of map-based verification and position claim overhearing. Map-based verification assigns a plausibility value to the received beacons by comparing the location to a road map. Position claim overhearing can be applied in (geo)routing scenarios: by comparing different overheard packets and their destinations, overheard packets can provide indications of a false position in the past. As the authors note, this check is not very strong, and may generate false positives.

Another approach is to actively detect positions through the usage of cells, as done in [95]. Cells are circular areas, which in the case of this paper are position-based using GPS coordinates. The cells are given a radius of 100 meters, which matches the range of the radar the authors assume each vehicle has. Vehicles associate themselves with a cell by sending their identity to the cell leader; if there is none, that vehicle becomes cell leader. Cell leaders are tasked with verifying the positions of all other vehicles in the cell; these members check the transmissions of the leader with their information. Vehicles can challenge the current cell leader and replace it, if that vehicle is closer to the center and approaching the leader. The leader is also responsible for forwarding location information from within the cell to other cells using cell routers, which is secured using an unspecified monitoring approach. Because the leaders and routers are the only ones authorized to transmit positions of vehicles to other cells, this prevents an attacker from falsifying his position towards vehicles outside the cell. To protect against Sybil attacks, similarities between received information and the radar are computed (i.e., a consistency mechanism), which is compounded with history that is kept per vehicle and uses predictions, similar to the Kalman filters proposed to be used by Stübing et al. [87] (which we discuss later in this section). To isolate vehicles, the authors use three tables that keep trusted, questionable and distrust tables. The authors also list a number of possible attacks against their scheme. However, notably, they do not discuss the enormous bandwidth requirements of constantly changing leaders and routers, as well as the interactive protocols they describe. At one point, the paper discusses acknowledgments of messages, which implies the use of unicast and thus additional overhead, but even without that the resource requirements are significant for a scheme with potential attacks.

Vehicle behavior analysis and Evaluation scheme (VEBAS), was presented by Schmidt et al. [82] as an attempt to build a full system up out of behavioral, plausibility and trust-based mechanisms. VEBAS allows the local detection of unusual vehicle behavior. Each vehicle analyzes all messages received from neighboring vehicles to detect possible misbehavior. Schmidt et al. analyze the content of messages using different modules, which are shown in Table 2. In this, most mechanisms are plausibility checks, with either positive (meaning the mechanism states the information is correct) or negative (information is false) results. Some mechanisms analyze the behavior of a vehicle in the classical sense (i.e., behavioral mechanisms) while other mechanisms also make use of physical models. As an example of behavior-based mechanisms, the authors used a maximum beaconing frequency threshold. If a vehicle sends messages with an unusually high frequency, the behavior can be regarded as possible attack. The other mechanisms discussed by the authors are plausibility-based mechanisms, some of which are re-used from earlier work. Finally, the authors show a way to combine the output of the different detection mechanisms, the authors proposed to use 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. 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.

	Mechanism	Description
`MA+`	Movement analysis	Checks for valid average velocity, acceleration, and heading.
`<X>PP`	Sensor-proofed position	Captures sensor validation mechanisms, such as Lidar.
`MDM`	Minimum distance moved	Proofs that vehicles have moved during a certain time to detect static roadside attackers.
`ART`	Acceptance range threshold	Detects unreasonably high distance to a position claimant.
`MA-`	Movement analysis	Negative ratings occur when claimed movement is impossible.
`MPP`	Map-proofed position	Returns negative values if a claimed position is not found on the road map.
`SAW`	Sudden appearance warning	Vehicles should first appear at the boundary of the reception range and then move inwards; this module reports deviations.
`MBF`	Maximum beaconing frequency	Detects violations of common maximum beaconing frequencies.

Table 2: Misbehavior detection mechanisms used in [82]. The first group provides positive ratings for nodes; the second group provides negative ratings.

Kalman filters As discussed in Section 4.1.2, the advantages of node-based mechanisms are strengthened when pseudonyms can be resolved, or at least linked, in the local vicinity of a vehicle. The Kalman filter approach does exactly this, relying on data transmitted by each node instead of a cryptographic mechanism that imposes additional restrictions on the system.

The authors of [87] and [45] take a different approach: they 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 the Kalman filter allows the accurate prediction of movement even under the influence of errors. As a result, the 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. Their scheme explicitly does not distinguish between malicious and faulty nodes, instead aiming to detect any misbehavior. This work has

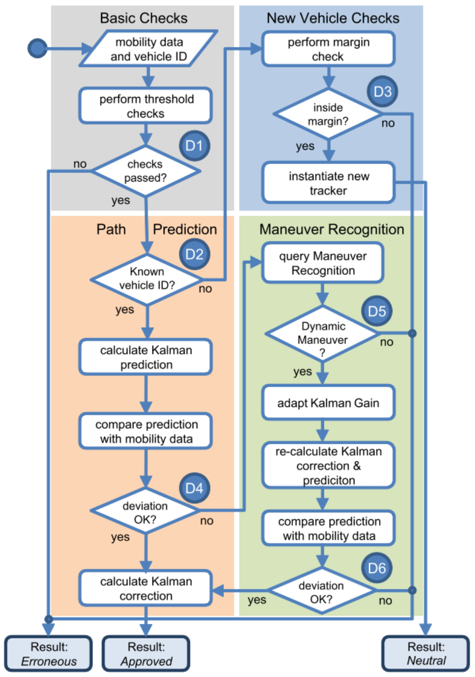


Figure 8: Figure from [86] explaining their detection method surrounding usage and updating of Kalman filters.

been exploited by more recent work to verify message sequences and illustrates the trade-off between security and privacy. 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 (similar to physically following it by driving behind it).

Building on this work, [86] proposes a second stage in the verification process. This verification process will support the Kalman filters in situations where they are less effective, 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. This detection is performed 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. This entire process is shown in detail in Figure 8.

Post-event detection The authors of [30] design a novel type of misbehavior detection scheme, aimed at the post-crash notification (PCN) application. Rather than proposing to detect misbehavior as it occurs, the authors propose to detect the misbehavior after-the-fact by observing the response of the driver. By reporting this confirmed misbehavior, it is possible to expel misbehaving vehicles from the network. Although many mechanisms allow detection to be performed or produced after the event, this mechanism is unique in that it relies explicitly on the driver for detection. Detection is performed by estimating two trajectories from the time of reception of the PCN until the vehicle reaches the location of the claimed event. These trajectories describe driver behavior in the case where there was indeed an incident, and regular driving behavior. By comparing these against the actual trajectory that the driver drives, one can estimate the actual situation and determine whether the PCN was legitimate. In addition, their scheme allows to extract the cause of the event using what the authors call root-cause analysis. This allows the receiver to detect which part of the message was falsified: the position information or the entire message. The scheme also has disadvantages

– it relies on the fact that driver behavior can be modeled with sufficient accuracy, and if the attacker can affect the measurements inside his own vehicle, he can still generate false reports. As discussed, the attacker model in our survey allows the attacker to control his vehicle completely. Nevertheless, if the reports are signed with a long term certificate, and falsified reports can be detected and distinguished from errors, this detection mechanism is still useful.

4.2.2 Consistency

Consistency-based mechanisms look at sequences of packets 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 combine information from several vehicles into aggregates and to deal with inaccuracies, which may occur when aggregation mechanisms are used.

Direct checking 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).

Golle et al. [31] 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 cITS. 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 cITS, 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 cITS is limited. In addition, it seems to be assumed that the attacker is a protocol-adhering participant, e.g., there are no attacks on routing or on lower communication layers (such as selective jamming or altered transmission power).

Leinmüller et al. [51, 54] describe a position verification mechanism that bases on a number of different algorithms (also called sensors), each of which attempts to detect malicious or selfish behavior. The position verification mechanism determines a trust value for each vehicle, but the focus of the paper lies on the sensors. The authors propose sensors based on either consistency or plausibility (called cooperative and autonomous sensors respectively in the paper); we discuss the consistency sensors here, and the plausibility sensors in Section 4.2.1. The cooperative sensors are based on neighbor tables and position beacons to avoid the requirement of dedicated hardware. First, pro-active exchange of neighbor tables can include positions or only include logical links between nodes. In both cases, beacons are checked against received neighbor tables by comparing the claimed positions for a particular node in the beacon and the table. When the tables do not include positions directly, nodes can extrapolate information using the maximum transmission range (using the accepted range threshold, see Section 4.2.1). Second, reactive position requests can be used as a more bandwidth-efficient sensor than periodic neighbor table exchange. These requests are sent when an unknown vehicle M is encountered; a vehicle knows the position of its neighbors, and selects a subset of them as either rejector or acceptor, based on whether the neighbor is in transmission range or not. It then sends its request to this subset of neighbors, asking for the position of M. Neighbors that do not know

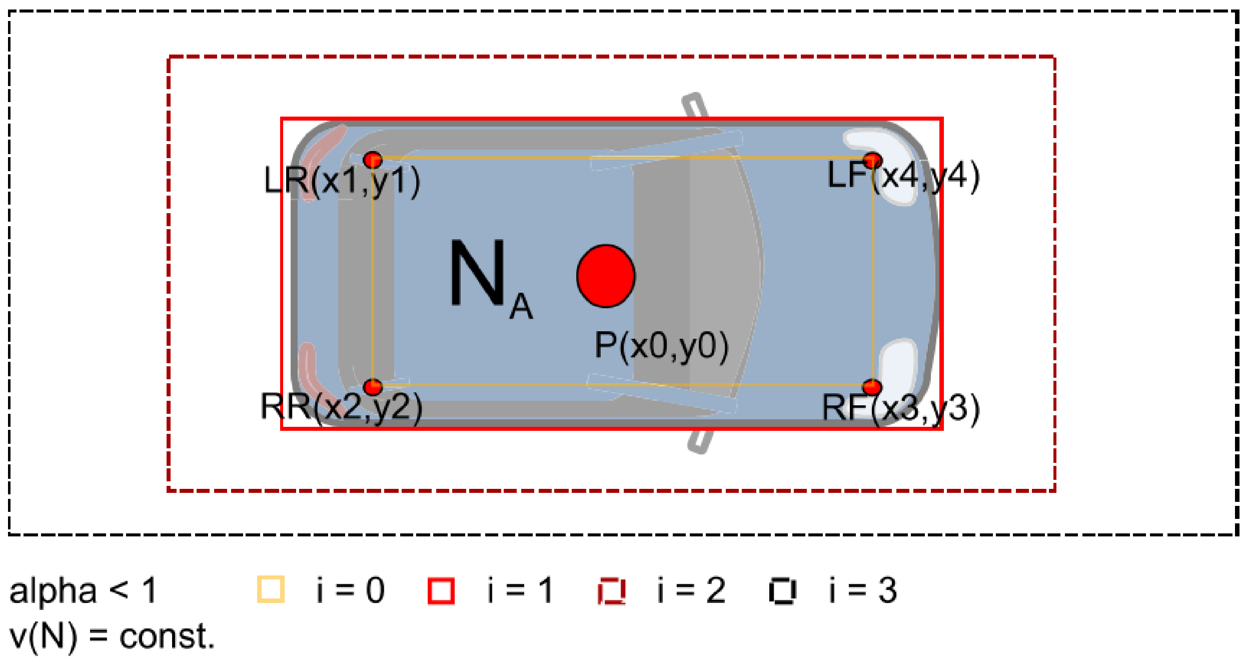


Figure 9: Figure from [8] showing the different rectangles used to model the vehicle in the presence of GPS errors.

M will respond with a corresponding message; others will respond with a position. The sender can then compare the responses with the expected responses. Both of these mechanisms rely on an honest majority, but are capable of dealing with noisy sensor data for those honest nodes. The mechanisms can also cope with network loss, allowing the mechanisms to cope with limited attacks on lower layers.

In [8], 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 discuss in Section 4.2.1) 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, as shown in Figure 9. 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.

Data-centric security mechanisms have also been applied to secure aggregation, as proposed by [18]. Aggregation is here motivated as a means to make cITS more feasible. Due to high potential bandwidth requirements, especially for traffic efficiency applications that require information from large areas, aggregation will be essential to efficiently disseminating information through the cITS on a large scale. However, security is an unresolved issue in aggregation: in particular, aggregation aims to remove redundancy of information, which increases the challenge of applying data-centric security mechanisms. The authors specify a framework using fuzzy reasoning to detect that an attack is in progress with high accuracy. As input for this reasoning process, clues attached to each aggregate are used, which are atomic, unaggregated observations signed by their original senders. These clues provide the necessary integrity that is typically hard to obtain in secure aggregation. Using fuzzy logic to determine confidence in data, rather than trust in nodes, is a main advantage of the scheme. The attacker model in this paper considers an attacker that is specifically aimed at compromising aggregation mechanisms, rather than a general attacker that also exploits lower layer and Sybil attacks.

Sybil attack detection Because of the honest majority assumption that is typically made to enable consistency and node-centric mechanisms, Sybil attacks must be prevented for the schemes to be effective. There are two types of approaches that can be implemented as a consistency mechanism; either independently performed by message exchanges or using (partially off-line) support from RSUs.

Grover et al. [32] perform Sybil attack detection by comparing neighbor tables of several vehicles over time. The reasoning behind their work is that the fake identities of the attacker must always be in the vicinity of the attacker in order to be able to the necessary local majority the attacker wants to achieve. Therefore, they must frequently appear in the neighbor tables of legitimate vehicles as a group, while legitimate vehicles will not form such a group over time. Because network topology is highly dynamic in cITS, the authors distinguish between uni-directional and bi-directional communication links (called physical and communication neighbors, respectively). We note that while this detection mechanism presents a novel idea that may be effective to limit Sybil attacks in duration, the communication overhead is significant, as is the latency for detection. In addition, certain traffic scenarios cause the assumption of recurrence to become unrealistic, e.g., in traffic jams, which increases false positives or increases detection latency. Most importantly, it does not protect against Sybil attacks that have a short duration, which however may be sufficient to attack a variety of information dissemination mechanisms, such as traffic jam detectors.

Similarly, the authors in [94] focus on discussing the verification of location data by using additional vehicle sensors that allow detection by line-of-sight, so-called eye-devices, in order to verify received messages (which are referred to as coming from ear-devices). The authors propose that each vehicle should have a variety of radars and cameras to detect other vehicles. It is assumed that 85% of all vehicles is honest, and that GPS data can be trusted if the information corresponds to the own location combined with the data from eye-devices. However, eye-devices only work for line-of-sight; thus, the authors allow vehicles to request eye-device confirmation from vehicles on the oncoming traffic. Similarly, neighbors can be queried, although oncoming traffic is preferred. The authors express the eye data through vectors relative to the current vehicle (for speed and location). The authors then go on to define local security as verification by checking against local sensors, including ear-devices (messages from different vehicles). The authors claim that local security can be used to obtain global security, reasoning that this is analogous to greedy algorithms. However, we note that greedy algorithms are heuristics, which by definition do not necessarily provide a global solution. As the paper lacks any further proof of a globally optimal solution, it is likely that their mechanism can achieve at best a local solution. In addition, the authors do not seem to consider errors in the obtained locations and other sensor data, further weakening their claim. Finally, in low density scenarios the attacker may be able to exploit the mechanism to cause false detections using specially crafted messages, although this is a fairly obscure scenario.

Centralized detection Due to good detection performance achieved by some RSU-based detection mechanisms, as well as the significant amount of data available, many authors suggest that detection should be reproducible in a central location. This provides advantages such as quick revocation; some authors even suggest detection should occur completely in such a back-end system.

In [6], 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.

Data 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.

VARM [78] is an example of this idea, which directly applies data mining techniques to misbehavior detection. The authors propose a data mining-based mechanism that dynamically derives association rules from received data using a data-structure called the Itemset-Tree. Association rules express correlations in a data set, and are a basic concept in data mining. By extracting these association rules, the node thus infers information from received messages that represent the expected behavior of senders. This knowledge can express hidden information that represents local road conditions, without the need to list all such scenarios and develop rules or models for them by hand. Their core drawback is that they may not generalize, because correlation does not imply causation. Another issue is that the paper does not extensively study the bandwidth requirements posed by their scheme, nor is latency or detection rate the main target of the study. We note that data mining is typically applied in scenarios where latency and computational resources are not an issue, and these techniques may not provide sufficient performance. Nevertheless, the application of data mining is a novel idea that can combine elements from both data-centric and behavioral misbehavior detection; this work provides a good starting point for applying data mining techniques to misbehavior detection. VARM is considered to be a consistency-based misbehavior detection mechanism because the authors specifically focus on temporal relationships between events received from many different vehicles, rather than verifying individual vehicles.

Grover et al. [34] have also proposed a machine learning-based method to detect misbehaving vehicles. Grover et al. have implemented several attacks on routing and several attacks on message content, in particular a Sybil attack and position forging. They use machine learning to learn the features of legitimate and misbehaving messages, based on models of both. The authors defined several features and used a network simulator to generate traffic; the features of this traffic was used to train several classification algorithms. There are two classes of features – one used for detecting position falsification and Sybil attacks (similar to many data-centric mechanisms) and one class that considers various temporal aspects and delivery ratios. The generated network traffic was created using a two-way, multi-lane highway scenario. The authors used the values for each of these features as input for several classification algorithms from machine learning, contained in the WEKA toolset [37]. The best results were obtained using the Random Forest [9] and J48 [70] algorithms. Figure 10 shows the overall architecture of the proposed mechanism. In a later paper [33], they improved on their previous work by replacing the single classification algorithm with a number of classification algorithms. After classification by each of the algorithms, a majority decision is used to decide whether a given situation is considered part of an attack. In both papers, specific implementations of attacks are used to evaluate the system; these implementations are used to generate data for both the training and testing set. Both of these works rely on specific implementations of attacks and a specific scenario. This makes it very unclear whether the results can apply generally. In addition, the attacker model is limited to the specific attacks they describe, and it is not clear whether they can detect combinations of these attacks.

4.3 Overview

Now that we have provided an overview of different types of mechanisms that exist in the literature, we summarize these mechanisms in Table 3. The table contains the most important pros and cons for each,

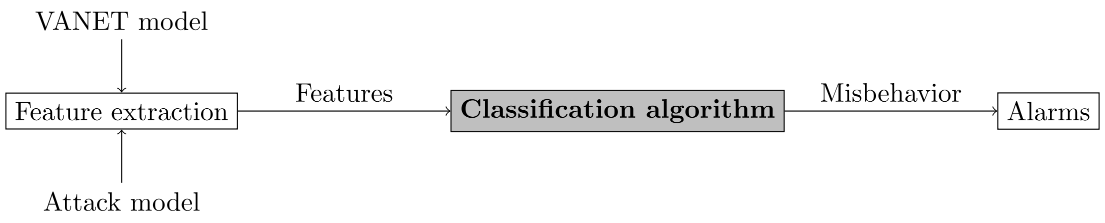


Figure 10: Machine learning architecture proposed in [34].

as well as a qualitative analysis of the scope of detection, required resources, generalizability, security and impact on privacy. The scope of detection tells us whether a mechanism performs local (L), cooperative (C) or back-end (B) detection, connecting back to the discussion on detection scopes in Section 3.3. The required resources field in the table gives an intuition of how much bandwidth, computational resources and memory is required; highly efficient mechanisms are marked with ⊕, while expensive mechanisms are marked with ⊖. Good generalizability (⊕) means that the mechanism may be transferred and applied to other domains relatively easily, a topic we discuss in detail in Section 5.3. With the security column, we illustrate how easily a mechanism can be manipulated: a mechanism subject to certain attacks is marked with ⊖, while a mechanism with a limited attacker model (compared to the model discussed in Section 3.1.2) is marked with ○and a good scheme is marked with ⊕. Note that this security column does not take into consideration the detection accuracy, but only the security of the detection mechanism. The last column, privacy, indicates the necessary type of linkability, as discussed in Section 3.4; full linkability (F), explicit linkability (E), implicit linkability (I) or no linkability (N). When the issue is not addressed in the paper, we use a question mark.

Table 3: Mechanisms overview

Mechanism	Pros	Cons	Scope[1]	Resources	Generalizability	Security	Privacy[2]	$\sum$
Node-centric / behavioral (Section 4.1.1)								
Example	—	—	–	⊕	○	⊖		
VEBAS [82]	Proposed plausibility and behavioral mechanisms are self-sufficient.	Broadcasting of votes requires honest majority assumption.	L,C	⊕	○	⊕	F	⊕
Watchdogs [40]	Low implementation complexity.	Only detects malevolent forwarding. Vulnerable to attacks, as discussed in their paper.	L	⊕	⊕	⊖	?	⊖
Correlation-based jamming detection [38]	Few known mechanisms for adaptive jamming detection.	Cannot identify or circumvent the attacker.	L	⊕	⊕	⊕	N	⊕
Node-centric / trust-based (Section 4.1.2)								
Dissemination with growth-codes [11, 46]	Simple and relatively efficient.	New nodes may have problems decoding older messages, weak attacker model.	C,B	⊕	⊕	⊖	F	⊖
Z-smallest probabilistic counting [41]	Strong protection against inflation attacks. Relatively efficient broadcast scheme.	Enforces a specific message format. The attacker model is weak.	C	○	⊕	○	E,I	○
Dynamic trustworthiness evaluation [74]	Can reuse results of the mechanisms.	No decision logic works best in all situations.	L	⊕	⊕	○	E	○
Bayesian logic-based [71]	Computes suspiciousness instead of trustworthiness; exploits Bayesian logic to derive suspiciousness.	Likely marks unusual but legitimate events, such as accidents, as suspicious. Requires knowledge of pre-computed conditional probabilities for all possible messages.	L	⊖	⊕	⊕	E	⊖
Cooperative trust building [52]	Introduces forwarding of simple detection results and employs trust transitivity.	Does not consider additional bandwidth requirements.	C	○	⊕	○	?	⊕
CMAP [39]	Secure without HSM, high verification efficiency.	Relies on weak attacker model, relies exclusively on proactive security, expensive RSU deployment and group maintenance.	C	⊕	⊕	⊖	E	⊖

Table 3: Mechanisms overview (continued)

Mechanism	Pros	Cons	Scope[1]	Resources	Generalizability	Security	Privacy[2]	$\sum$
Dynamic threshold trustworthiness [65]	Maximizes the confidence in messages until response is required.	May fail against targeted attacks designed specifically against this mechanism.	C	⊕	⊕	⊖	?	○
Certainty of Event [48]	Intuitive representation, combines security and safety aspects	Highly application-specific; relies on verified positions	C	⊕	⊖	○	E	⊕
LEAVE [73]	Aggregation improves the quality of detection and reduces latency.	Mechanism is prone to Sybil attacks.	C	○	⊕	○	?	○
SLEP & PRP [100]	Holistic approach, integrating cooperative and back-end detection.	Suicide could be prone to Sybil attacks, if an attacker guesses the right pseudonym change time.	B	○	○	○	E	○
OREN [5, 4] [76]	Game-theoretic evidence. Integrates issuance of pseudonyms as incentive system. Pseudonyms are not linkable.	Mechanism could be prone to Sybil attacks by financially capable attackers.	B	⊕	○	○	N	⊖
Hash-based linking of pseudonyms [97]	Solves issues with Sybil attacks.	RSUs can not be seen as ubiquitous or fully trustworthy; Strong privacy impact.	B	⊕	○	⊕	E	○
MisDis [96]	-	Lack of privacy compared to central authority, requirement of determinism, prohibitive size and cost	L,B	⊖	○	⊖	E	⊖
P^2DAP [98]	Strong detection/attacker model, potentially efficient approach	No privacy to central authority; event type-based; infrastructure-dependent detection	B	⊕	○	⊕	E	○
Urban path-based [15]	fast, computationally efficient detection	infrastructure-dependent, privacy issues, vulnerabilities, potential false positives	C	⊖	⊖	⊖	?	⊖
Footprint [14]	Reasonable level of privacy without high cost; useful for authenticating events	infrastructure-dependent; geared only towards Urban networks; full trust in all RSUs; potential false positives due to commuters	C	○	⊖	⊖	E,I	⊖
Data-centric / consistency (Section 4.2.2)								
Attacker modeling [31]	Structured and general approach.	Requires complex reasoning techniques on every vehicle.	C	⊖	○	○	I	○

Table 3: Mechanisms overview (continued)

Mechanism	Pros	Cons	Scope^1	Resources	Generalizability	Security	Privacy^2	$\sum$
Proactive cooperative sensors in [54]	Simplicity, capable of dealing with noisy data, powerful in dense scenarios.	Risk of Sybil attacks, risk of reduced reliability due to modified information, high bandwidth cost.	C	⊖	○	○	?	○
Reactive cooperative sensors in [54]	Simplicity, capable of dealing with noisy data.	May not work well due to high theoretical range, requires unicast message exchange.	C	○	○	⊕	?	⊕
Collision detection-based approach [8]	Good detection results under reasonable assumptions.	Requires fixed transmission power	C	⊕	⊖	⊕	I	⊕
Resilient secure aggregation [18]	Efficient approach to secure aggregation	only works for aggregation purposes	C	○	⊖	⊕	?	⊕
Neighbor-sets over time [32]	Novel ideas, prevents long-term Sybil attacks.	False positives possible; no prevention for short-term Sybil attacks.	C	⊖	○	○	?	○
Active position detection [94]	Dedicated hardware for secure positioning is highly effective; new cooperative mechanism.	High cost due to hardware requirements, privacy is not considered, can be attacked through Sybil attacks.	C	⊕	⊖	⊖	?	⊖
Back-end detection [6]	Uses global knowledge to improve/confirm detection	Potential privacy concerns; potential high cost at the back-end	B	⊖	⊕	⊕	E	⊕
VARM [78]	Novel approach, based on well-researched area	Feasibility and mapping of rules to conclusions is unclear.	L	⊖	⊕	○	?	⊕
Machine-learning-based detection [34] [33]	Machine learning allows easy generalization.	Prone to well planned attacks, because the classification algorithm could accept gradual changes. Decisions may not be transparent. Tested against specific implementations of attacks.	L	○	⊕	○	?	○
Data-centric / plausibility (Section 4.2.1)								
Plausibility Validation Network [58]	Fast and efficient, allows relations between plausibility checks	Insufficient as the only means to detect attacks	L	⊕	⊖	○	I	⊕
[54] and [51]	Efficient	Limited detection capabilities	x	x	x	x	x	x
Signal Analysis [35]	Thorough analysis of RSSI-based estimation	Potentially infeasible with GPS errors	L	⊕	⊖	○	N	⊕

Table 3: Mechanisms overview (continued)

Mechanism	Pros	Cons	Scope[1]	Resources	Generalizability	Security	Privacy[2]	$\sum$
Transmission time-based [79]		Unrealistic assumptions	x	x	x	x	x	x
Kalman filter-based [87, 45]	Highly efficient tracking of vehicles.	Requires many matrix computations for every neighbor. Could become less feasible if additional privacy-protection is introduced.	L	⊕	⊖	⊕	I	⊕
Extended Kalman filters [86]	Improvement upon above	Stronger privacy concerns	L	⊕	⊖	⊕	I	⊕
Cell-based [95]	Comprehensive approach	Cell-based approach, which is unrealistic in general VANETs, other unrealistic assumptions	L,C	⊖	⊖	⊖	?	⊖
Enhanced Position Verification [93]	Works well in ideal situations	Limited infrastructure dependence, network structure dependence, potential for attacker manipulation	L(,C)	⊕	⊖	⊖	?	○
After-the-fact verification [30]	Highly useful for reporting	No preemptive detection, each application requires accurate predictions	L	⊕	○	⊕	?	⊕

[1]) L: local, C: cooperative, B: backend

[2]) **TODO** explain linking options

5 Solved and Open Challenges

Starting with an overview of the current cITS ecosystem, as well as ongoing standardization efforts, we have provided an extensive categorization of different misbehavior types and categories of misbehavior detection mechanisms. In this section we discuss the solved and open challenges for security in cITS. In addition, we provide a discussion of ways to transfer the security mechanisms to other fields.

Related work has established many ways to implement pro-active security, which can prevent unauthorized entities from participating in the network. Specific mechanisms are currently being standardized, as discussed in Section 2. As a result, attackers simply using commodity hardware can be excluded from the network. Likewise, vehicles that were detected repeatedly attacking the network can be excluded this way. Therefore, pro-active security will play an important role in securing future deployments of cITS.

However, pro-active security will never be able to achieve perfect security, as we have discussed in Section 3. Depending on the lifetime of PKI-issued certificates, pro-active security mechanisms need to be complemented with revocation systems to guarantee the reliability of the system. Although the problem of distributing revocation information is well-studied (e.g., [64, 36]), the mechanisms that determine which certificates should be revoked are not. We claim that these mechanisms require misbehavior detection: selecting which certificates should be revoked requires that the attacker can be detected. In addition, due to the time-critical nature of systems like cITS, as well as in many other types of CPS, this detection of attackers must occur in an automated fashion. This leads to node-centric misbehavior detection mechanisms, whose majority focus on identifying the attacker. However, node-centric mechanisms often require the vehicles to be uniquely identifiable, which is questionable from a privacy perspective. Therefore, we need to consider scenarios where vehicles have multiple valid certificates (i.e., pseudonyms). Because of the fact that pseudonyms limit the applicability of node-centric misbehavior detection, we need to complement it with data-centric detection mechanisms.

Moreover, pro-active security is costly in terms of computational and bandwidth overhead, because signatures and certificates have to be attached to and verified for a large fraction of all messages, which leads to an increase in size up to twice as much as before [26] and significant computational overhead [66]. In addition to these critical issues, pro-active security carries with it high costs for administration, secure storage and revocation; although these may be solved by significant investments, this is not a desirable solution. Newer proposals consider omitting certificates for certain messages. However, omission schemes open attack vectors in case an attacker is aware of the omission pattern and therefore can use it to her advantage. As already identified in related work [83, 28], the trade-off between bandwidth usage and resulting security is a very challenging one, as pointed out in Section 5.2. Reactive security in the form of misbehavior detection components can help to complement the effect of such omission schemes. First, they can analyze message content and derive confidence in message correctness even if signatures and/or certificates are omitted, although some schemes may not be effective (as they require protection against Sybil attacks). Second, they can still identify misbehavior even if correct signatures are attached.

In this survey, we have reviewed the existing work in reactive security mechanisms and classified them as node- and data-centric mechanisms. This classification is useful, as the two types are mostly complementary: we have pointed out that many of these schemes can be used as separate sources of information on misbehavior. To summarize our results, we will discuss a number of reoccurring patterns and trends in existing work and outline specific open issues. We then generalize this to other CPS in Section 5.3.

5.1 Reoccurring patterns

In our discussion of the mechanisms, we often pointed out potential attacks tied to other challenges, which are solved by different mechanisms. Indeed, many mechanisms in our discussion are complementary, and here we discuss several recurring patterns of approaches taken by these mechanisms.

Physical models A number of mechanisms use physical models to detect misbehavior [54, 8]. These models allow a detection mechanism to incorporate specific knowledge about the driving process of vehicles. The complexity of models ranges from very simple (e.g., cars cannot drive faster than 500 km/h, two cars cannot occupy the exact same position) to very complex models (e.g., analysis of acceleration and deceleration behavior, movement prediction including turn probabilities, correlation of vehicle positions to street maps). These models may also include the sensors of the observing vehicle. The correlation with such models is a very interesting approach, because it often works locally and does not depend on correlation of data from several vehicles. Therefore, detection components based on physical models should be included in a misbehavior detection system.

Signal properties Besides physical models targeted at the process of driving itself, physical properties of the wireless channel (i.e., signal properties) are another important tool for detection of spurious messages. The typical wireless reception range can be estimated, and moreover, WiFi hardware is able to measure the strength of a received signal. When vehicles synchronize their clocks with GPS, time of flight measurements can also be used. All these observations can be used to detect messages which are, for instance, received with a high signal strength but claim to originate from a position far away. In addition, there have been proposals to include additional hardware (typically additional antennae, see [77]) to effectively perform more advanced signal analysis to determine the direction from which the signal originates. However, the effectiveness of signal-based mechanisms is somewhat disputed due to the high mobility, cost constraints and potential inaccuracy [51, 35]. This will be especially challenging when combined with newly proposed MAC-protocols that exploit the additional spatial re-use enabled by transmit power variations, such as decentralized congestion control in ETSI [23].

Machine learning techniques A problem that many mechanisms face is how to interpret locally collected data in order to find patterns, or how to merge results from a set of different misbehavior detectors into one coherent output. To solve these problems, a number of schemes adapt ideas from machine learning. Often-employed approaches include decision trees, Bayesian inference or Dempster-Shafer theory [74, 71, 78]. These approaches can provide useful tools to analyze data and derive certain interesting features that might point at attacks. Some reputation systems, such as those proposed in [4], already use some of these ideas to manage node reputation, but these typically only focus on node-centric aspects, such as how to maintain the reputation system and how to protect it against attacks. In addition to providing misbehavior detection for vehicles, these node-centric approaches may be later used to perform an after-the-fact analysis, which in turn can be used to verify detected misbehavior.

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 [6] and [98] 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.

5.2 Open Issues

Even though many specific aspects of misbehavior have been addressed, there are still a number of open issues. We discuss these issues based on our survey of the state of the art and the mechanisms discussed in Section 4. These issues are challenges regarding the detection itself – dealing with conflicting data, for instance by discarding or hiding messages, error correction, reporting and so forth is an orthogonal issue.

Thresholds for detection An important problem in any intrusion detection system is its configuration; after which threshold is a message malicious? This question is also an important challenge in misbehavior detection, because high false positive or false negative rates can cause significant problems or even cause the system to be rejected entirely. Especially for cITS, where direct reporting to an expert through warnings is not always possible, this needs to be addressed. As a related issue, the distinction between erroneous and explicitly malicious messages would be useful, although this may not be feasible. Additionally, because of the data-centric nature of many detection mechanisms, the question of when information should be merged is significant, and this may affect detection performance directly.

Identification of misbehaving nodes We note that while data-centric mechanisms are superior at detecting conflicting data and identifying which data is malicious, they may not be able to identify the attacker directly. As for many applications, it is more important to have accurate data than it is to identify attackers to mitigate possible harm through attacks, node-centric mechanisms that identify attackers are still useful. They can be used to perform local or global revocation, thereby limiting both the amount of attackers in the network and the impact they may have on the network. As an example take the fake traffic jam scenario from Figure 2b in Section 1.3. Although a data-centric mechanism may detect that the reported traffic jam is fake, without proper identification of the sender, a local revocation is not feasible. Finally, identifying which data belongs to which vehicle may provide additional input for node-centric mechanisms, for instance to assign a credibility to information sources.

Voting, pseudonyms and Sybil attacks Many node-centric mechanisms 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 2.4. Therefore, an important open issue is to build a scheme that performs voting using pseudonyms, while still providing the necessary resilience against Sybil attacks. A simple solution would be to require that a vehicle can only use one pseudonym at a time, but this assumption is disputable both from a practical and a privacy point of view. Although some work has already been done (see Section 4.1.2), it is an open challenge to design a voting scheme where all participants can use arbitrary pseudonyms while still allowing to reveal multiple votes by the same vehicle. One possible solution for this challenge is to assume limited processing capabilities for the attacker and employ a cryptographic primitive called Proof-of-Work (PoW). As the name implies, this primitive allows a vehicle to prove that it has performed some amount of computation, and it is traditionally employed to hinder denial of service attacks. This works because the computation becomes exponentially harder and cannot be precomputed. One proposal to use PoW mechanisms for cITS is [62], which requires PoW for evidence that an event actually occurred. The problem with these approaches is that they pose significant computational overhead, and thus cannot work in all situations.

Linkable messages While the aforementioned mechanisms are beneficial for detecting misbehavior, they might have negative impact on the location privacy the users. As discussed above, pseudonym schemes are employed to prevent attackers from collecting location traces of specific vehicles over a longer time. The standardization efforts aim mainly at preventing privacy impact on a larger scale: the goal is to prevent attackers from efficiently following many participants, given a few stationary attacker-controlled network nodes. This is opposed to attackers that track a single vehicle with a moving node – this is, after all, equivalent to physically following a vehicle. It has been shown by privacy research [92] that it is possible to track vehicles in this manner, which is indeed necessary for certain safety applications, such as those that rely on path prediction. This functionality is provided by employing movement models for vehicles to predict future positions and correlate them with current positions. Such linking can provide the necessary information, which can not only enable cITS applications, but also misbehavior detection. Indeed, several data-centric mechanisms, such as [86], already exploit this approach. However, it is an open challenge to assess the exact privacy impact of these mechanisms and their full potential for misbehavior detection. In particular, the reliability of these mechanisms can subsequently have a large impact on accuracy, which an attacker could exploit. We remark that mechanisms relying on these approaches may be defeated by future, more powerful privacy mechanisms.

Cooperative misbehavior detection As long as vehicles or other entities only analyze locally received data to detect misbehavior, trust in the derived opinions is not a big issue. That is, we can assume that vehicles trust the mechanisms running locally. Even though the decisions reached might be associated with uncertainty, that uncertainty is still known. Once vehicles exchange reports with other vehicles, the situation is different. A misbehavior report received by another vehicle is basically just another item of information, like any other information item received from remote vehicles. Therefore, 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. Other approaches propose to solve the issue by so-called suicide schemes where vehicles reporting misbehavior of other vehicles also exempt themselves from the network (see Section 4.1.2). However, another large share of related work ignores the issue of trusting cooperative misbehavior reports. These schemes assume that reports from other vehicles are trustworthy if they carry correct signatures. An ideal solution for cooperative misbehavior detection has not been presented yet.

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.

5.3 Applications beyond cITS

Now that we have studied and classified the state of the art misbehavior detection mechanisms for cITS, we address the possible application of the ideas of these mechanisms to other cyber-physical system (CPS). Although many possible choices exist, we have chosen two domains that seem most appropriate: WSNs, due to their ephemeral and ad-hoc nature, and ICS, due to the potential for (especially) plausibility-based misbehavior detection mechanisms that we have identified.

5.3.1 Wireless Sensor Networks

Similar to cITS, WSNs are a type of network that is built up out of many different nodes with significant geographic distribution and connected through a wireless medium. Contrary to cITS, mobility is limited in a WSN, as sensors are relatively stationary entities. The main purpose of WSNs is to collect information in an efficient and cost-effective manner, using cheap throw-away devices that have a battery and a wireless transmitter. For example, WSNs are used to monitor the temperature of the great barrier reef [42]. For the lifetime of these networks, it is extremely important to monitor resource usage, particularly when transmitting messages, because transmission of messages is by far the most costly operation in terms of battery power. There are usually also one or more base stations involved in a WSN, to provide access to the sensor functionality and to perform more expensive operations [99].

There is already existing work that extensively surveys attacks and defensive techniques for WSNs, which we briefly discuss below. We discuss why these approaches cannot be generalized to CPS, while the mechanisms we have reviewed show significant potential to be generalized and applied to WSNs.

Zhou et al. [99] describe the WSN setting primarily as defined by resource-constrained sensors that perform both sensing and processing, while having a central authority that establishes the network. The network is mostly static and includes one or more base stations controlled by the authority. The main security challenges identified by the authors are the public nature of both the wireless medium and the protocols, in addition to the lack of (physical) surveillance of the nodes and finally and most importantly the extreme resource constraints. They also identify node compromise as one of the most challenging attacks on WSN, which we have similarly identified as applying to general CPS. Contrary to the cITS use case, WSNs typically also include confidentiality requirements. The authors then provide a thorough analysis of key management protocols, authentication and integrity, secure routing, intrusion detection and secure applications. Out of these security mechanisms, we focus on those relating to our discussions regarding reactive security: integrity and intrusion detection, which are the main issues that we see recurring in CPS.

Before discussing reactive mechanisms, we discuss some noteworthy pro-active security mechanisms from [99] that address integrity requirements. In particular, the survey discusses the usage of message authentication codes (MesACs) to address integrity (and authentication) in both symmetric and asymmetric settings; this is analogous to what we have seen in cITS and general networking applications. Both one-hop, multihop and broadcast authentication are discussed, notably including the TESLA adaptation for WSNs, called μTELSA, which has also been adapted for VANETs as TESLA++ [88]. The authors have pointed out that μTELSA provides delays that allow denial of service attacks, while asymmetric solutions are too computationally expensive, mainly due to the verification of the public key certificates involved. Finally, the authors explain a potential trade-off between message size and efficiency. The authors do not address the use of pseudonyms to protect location privacy, although privacy issues related to message content and routing are briefly addressed. This is because in WSNs, the individual nodes do not directly relate to persons, unlike vehicles in cITS. Finally we remark that TESLA-variants are exactly that which one would expect from pro-active security mechanisms – they protect the integrity of the message in transit.

As we have emphasized, protecting the message in transit is not sufficient when insider attacker need to be considered. The authors of [99] address the issue of node compromise in their section on intrusion detection. Although node compromise implies an outside attacker, the effects on the network are essentially the same; an attacker can transmit validly signed message. The only difference is that when a node attacks itself (as an insider in a cITS would), the node still has an interest in maintaining the functionality of the system, as opposed to node compromise, which may simply be intended to disrupt the system. The authors discuss several intrusion detection systems in the broadest sense of the term: their discussion includes the use of location-based keys (which is feasible due to the relatively static nature of the network), signal analysis to counter Sybil attacks (see also Section 4.2.1) and several mechanisms to counter selective forwarding and other routing attacks (typically variants of Watchdog mechanisms, see Section 4.1.1). One issue that has not been discussed in the context of cITS is that of node replication. Finally, the authors have discussed denial of service and jamming a type of attack that is extremely hard to detect, and secure base stations as another issue that is fairly specific to WSNs. The authors note that it is possible that base stations become compromised, even though they are generally assumed to be secure, followed by a discussion of preventive mechanisms for base stations. Similar assumptions exist in cITS for road side units (RSUs), but the detection of malicious RSUs is generally easier because they are not essential to a cITS the way they are for WSNs. Mechanisms employed for cITS could be used to address this challenge in exchange for higher computational overhead.

Of further interest is their discussion of (active) attacks, which surveys the most important types of attacks on WSNs and their countermeasures. We have already covered the issue of Sybil attacks and wormhole attacks; in addition to these, the authors discuss selective forwarding, node replication, and the rushing attack. Selective forwarding and the rushing attack are both attacks on routing protocols, where the attacker attempts to manipulate the traffic to either decrease its load, or to encourage forwarding of traffic for the purpose of analysis. The node replication attack, on the other hand, is similar to the Sybil attack in that it allows the attacker to create multiple instances (copies) of a compromised node. The solutions discussed to protect against these attacks use witness nodes to detect this attack, in a process that is comparable to the usage of watchdogs to detect the discussed routing attacks (including the wormhole attack). We note that these approaches are similar to node-centric mechanisms that we have surveyed, although their focus is more on efficient computation and exploit the relatively static and centralized network architecture that a WSN offers, compared to our discussion of schemes for the cITS use case.

Finally, as their section on open issues describes [99], detecting intruders remains a difficult task, and many mechanisms only focus on a particular attack. We claim that our survey provides further insight into what mechanisms exist, and we include several mechanisms that describe combinations of existing schemes. For future work, we believe a primary goal of the CPS research community should be to develop a more systematic approach to this problem. This approach can then be aided by our survey, which can be used to select different types of mechanisms to gain an overall insight of potential attackers in the network. This includes the transfer of mechanisms described here into the area of WSNs, in particular the recurring patterns we have identified in the previous section. For example, physical models that describe the behavior of a complex system monitored by a WSN can be used to verify the messages transmitted by individual nodes at a base station, or at an aggregating node.

5.3.2 Industrial Control Systems

this section will be extended after the Dagstuhl meeting Unlike cITS, ICS are a type of CPS that has historically grown out of the field of control systems, and the focus of these systems has primarily been safety, rather than security. Specifically, ICS 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, ICS 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 cITS may provide significant insights for new developments in ICS, for example through the application of physical models to detect anomalies with high accuracy.

6 Conclusion

In this survey, we have surveyed misbehavior detection in cITS. cITS are a promising type of networked system that connect vehicles, road-side units and back-end systems in order to achieve a safer, more efficient and more comfortable travel on roads. cITS are an instance of CPS, a type of system where interaction between the physical world and the cyber world is a central aspect. CPS have several unique challenges, including the critical usage scenarios, strong resource and cost constraints, and high scalability requirements. These challenges lead to new and strong security requirements, which lead to the development of misbehavior detection as a second, reactive layer of security on top of the first, proactive layer. In the case of cITS, this proactive security layer is the PKI, which enables the exclusion of attackers that do not possess key material. However, in highly constrained environments like CPS, some network nodes may be compromised to obtain key material; in particular, in cITS, an attacker may obtain legitimate key material by manipulating her own vehicle to send arbitrary messages. Therefore, reactive security in the form of misbehavior detection is a necessary tool to provide security in cITS. We have surveyed such misbehavior detection mechanisms in detail, and discussed their generalizability towards other CPS.

To this end, we have first defined the system model for cITS (Section 2) and the concept of misbehavior in these systems (Section 3). We then presented a classification of different misbehavior detection mechanisms that have been developed in the literature over the last decade, designed specifically for cITS. The classification consists of node-centric mechanisms, which use properties of a sender to detect malicious messages, and data-centric mechanisms, which primarily analyze the semantics of received messages. Node-centric mechanisms can be sub-divided in two further classes; behavioral mechanisms, where the receiver analyzes aspects like message frequency and conformance to standards, and trust-based mechanisms, which allow nodes to express trust in messages or senders directly. Data-centric mechanism can be divided in plausibility mechanisms, analyzing the semantics of a series of packets from an individual sender, and consistency mechanisms, which analyze the consistency of messages received from multiple different senders. The classification has been used to discuss a large number of different misbehavior detection mechanisms developed in the literature, the results of which are summarized in Table 3.

Although we have limited the survey of mechanisms to those designed for cITS, we believe that this survey offers a contribution to the wider field of CPS, as the novel aspects of mechanisms developed for cITS can be used to improve the security of other CPS. To take a first step towards this goal, we have analyzed the common aspects in Section 5 and discussed their application to other systems from the CPS-domain. We envision two broad paths for future work; for security in cITS, taking further advantage of the highly orthogonal nature of the four different classes of misbehavior detection we have presented is of vital importance, while the generalization of misbehavior detection mechanisms for cITS to general CPS will allow the field to advance without sacrificing security. For CPS, security is not optional: these systems control physical processes, many of which present a critical usage scenario where failure may cause extensive damage or loss of life.

Acknowledgements

We thank Anke Jentzsch and Hendrik Decke from Volkswagen AG, Germany, for their cooperation, feedback and fruitful discussions concerning this survey. This work was supported in part by the Baden-Württemberg Stiftung gGmbH Stuttgart as part of the project IKT-05 AutoDetect of its IT security research programme.

References

[1] S. Amin, G. A. Schwartz, and A. Hussain. In quest of benchmarking security risks to cyber-physical systems. *IEEE Network*, 27(1):19–24, January 2013.

[2] D. Antolino Rivas, J. M. Barceló-Ordinas, M. Guerrero Zapata, and J. D. Morillo-Pozo. Security on VANETs: Privacy, misbehaving nodes, false information and secure data aggregation. *Journal of Network and Computer Applications*, 34(6):1942–1955, Nov. 2011.

[3] N. Banerjee, M. D. Corner, D. Towsley, and B. N. Levine. Relays, base stations, and meshes: enhancing mobile networks with infrastructure. In *Proceedings of the 14th ACM international conference on Mobile computing and networking*, MobiCom '08, pages 81–91, New York, NY, USA, 2008. ACM Press.

[4] I. Bilogrevic, M. H. Manshaei, M. Raya, and J.-P. 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)*, pages 21 –30. IEEE, 31 2010-june 4 2010.

[5] I. Bilogrevic, M. H. Manshaei, M. Raya, and J.-P. Hubaux. Oren: Optimal revocations in ephemeral networks. *Computer Networks*, 55(5):1168 – 1180, 2011.

[6] N. Bißmeyer, J. Njeukam, J. Petit, and K. M. 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)*, pages 73–82, New York, NY, USA, 2012. ACM Press.

[7] 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, pages 162–167, ICST, Brussels,

Belgium, Belgium, 2011. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[8] N. Bißmeyer, C. Stresing, and K. M. Bayarou. Intrusion Detection in VANETs Through Verification of Vehicle Movement Data. In *Proceedings of the Vehicular Networking Conference (VNC)*, pages 166–173. IEEE, 2010.

[9] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[10] P. Brutch and C. Ko. Challenges in intrusion detection for wireless ad-hoc networks. In *Proceedings of the Symposium on Applications and the Internet Workshops*, pages 368–373. IEEE, Jan. 2003.

[11] 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*, pages 1–6. IEEE, 2008.

[12] M. Castro and B. Liskov. Practical byzantine fault tolerance. In *Proceedings of the Third Symposium on Operating Systems Design and Implementation*, pages 173–186, New Orleans, USA, February 1999. USENIX.

[13] D. J. Chaboya, R. A. Raines, R. O. Baldwin, and B. E. Mullins. Network intrusion detection: Automated and manual methods prone to attack and evasion. *Security Privacy*, 4(6):36–43, 2006.

[14] S. Chang, Y. Qi, H. Zhu, J. Zhao, and X. Shen. Footprint: Detecting Sybil Attacks in Urban Vehicular Networks. *Transactions on Parallel and Distributed Systems*, 23(6):1103 – 1114, 2012.

[15] 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*, pages 270–276. IEEE, June 2009.

[16] H. Debar, M. Dacier, and A. Wespi. Towards a taxonomy of intrusion-detection systems. *Computer Networks*, 31(8):805 – 822, 1999.

[17] A. P. Dempster. Upper and lower probabilities induced by a multivalued mapping. *The annals of mathematical statistics*, 38(2):325–339, 1976.

[18] S. Dietzel, E. Schoch, B. Könings, M. Weber, and F. Kargl. Resilient secure aggregation for vehicular networks. *Network*, 24(1):26–31, Jan. 2010.

[19] D. Djenouri, L. Khelladi, and N. Badache. A survey of security issues in mobile ad hoc and sensor networks. *Communications Surveys Tutorials*, 7(4):2–28, 2005.

[20] J. R. Douceur. The sybil attack. In P. Druschel, F. Kaashoek, and A. Rowstron, editors, *Peer-to-Peer Systems*, volume 2429 of *Lecture Notes in Computer Science*, pages 251–260. Springer Berlin Heidelberg, Mar. 2002.

[21] J. M. Estévez-Tapiador, P. Garcia-Teodoro, and J. E. Díaz-Verdejo. Anomaly detection methods in wired networks: a survey and taxonomy. *Computer Communications*, 27(16):1569 – 1584, 2004.

[22] 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.

[23] ETSI. Intelligent Transport Systems (ITS); Decentralized Congestion Control Mechanisms for Intelligent Transport Systems operating in the 5 GHz range; Access layer part. Technical Specification: TS 102 867, V1.1.1, July 2011.

[24] 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.

[25] 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.

[26] 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.

[27] 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, page 135138, New York, NY, USA, 2012. ACM.

[28] M. Feiri, J. Petit, and F. Kargl. Congestion-based certificate omission in vanets. In *Proceedings of the ninth ACM international workshop on Vehicular inter-networking, systems, and applications*, VANET '12, pages 135–138, New York, NY, USA, 2012. ACM Press.

[29] P. García-Teodoro, J. E. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers & Security*, 28(1-2):18–28, Feb. 2009.

[30] M. Ghosh, A. Varghese, A. Gupta, A. A. Kherani, and S. N. Muthaiah. Detecting misbehaviors in VANET with integrated root-cause analysis. *Ad Hoc Networks*, 8(7):778–790, Sept. 2010.

[31] 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)*, page 29, New York, NY, USA, 2004. ACM Press.

[32] J. Grover, M. S. Gaur, V. Laxmi, and N. K. 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)*, page 151, New York, NY, USA, 2011. ACM Press.

[33] J. Grover, V. Laxmi, and M. Gaur. Misbehavior detection based on ensemble learning in vanet. In P. Thilagam, A. Pais, K. Chandrasekaran, and N. Balakrishnan, editors, *Advanced Computing, Networking and Security*, volume 7135 of *Lecture Notes in Computer Science*, pages 602–611. Springer Berlin / Heidelberg, 2012.

[34] J. Grover, N. K. Prajapati, V. Laxmi, and M. S. Gaur. Machine learning approach for multiple misbehavior detection in vanet. In A. Abraham, J. L. Mauri, J. F. Buford, J. Suzuki, and S. M. Thampi, editors, *Advances in Computing and Communications*, volume 192 of *Communications in Computer and Information Science*, pages 644–653. Springer Berlin Heidelberg, 2011.

[35] G. Guette and B. Ducourthial. On the Sybil attack detection in VANET. In *2007 IEEE Internatonal Conference on Mobile Adhoc and Sensor Systems*, pages 1–6. IEEE, Oct. 2007.

[36] J. J. Haas, Y.-C. Hu, and K. P. Laberteaux. Design and analysis of a lightweight certificate revocation mechanism for VANET. In R. Shorey, A. Weimerskirch, D. Jiang, and M. Mauve, editors, *Vehicular Ad Hoc Networks*, pages 89–98, New York, NY, USA, 2009. ACM Press.

[37] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. In R. L. Grossman, O. R. Zaine, C. Aggarwal, and B. Goethals, editors, *Special Issue: Open Source Analytics*, volume 11 of *KDD Explorations*. KDD, July 2009.

[38] A. Hamieh, J. Ben-Othman, and L. Mokdad. Detection of radio interference attacks in VANET. In *IEEE Global Telecommunications Conference (GLOBECOM)*, pages 1 –5. IEEE, 30 2009-dec. 4 2009.

[39] Y. Hao, Y. Chengcheng, C. Zhou, and W. Song. A distributed key management framework with cooperative message authentication in vanets. *Journal on Selected Areas in Communications*, 29(3):616 –629, march 2011.

[40] J. Hortelano, J. C. Ruiz, and P. Manzoni. Evaluating the usefulness of watchdogs for intrusion detection in VANETs. In *IEEE International Conference on Communications Workshops (ICC)*, pages 1–5. IEEE, May 2010.

[41] H.-C. 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, pages 163–174, New York, NY, USA, 2011. ACM Press.

[42] C. Huddlestone-Holmes, G. Gigan, G. Woods, A. Ruxton, I. Atkinson, and S. Kininmonth. Infrastructure for a sensor network on davies reef, great barrier reef. In *3rd International Conference on Intelligent Sensors, Sensor Networks and Information*, pages 675–679. IEEE, Dec. 2007.

[43] IEEE. IEEE 1609.2: IEEE standard for wireless access in vehicular environments - security services for applications and management messages, 2013.

[44] S. International. Dedicated short range communications (DSRC) message set dictionary. Technical Report J2735, November 2009.

[45] A. Jaeger, N. Bißmeyer, H. Stübing, and S. A. Huss. A Novel Framework for Efficient Mobility Data Verification in Vehicular Ad-hoc Networks. *International Journal of Intelligent Transportation Systems Research*, 10(1):11–21, Aug. 2011.

[46] A. Kamra, V. Misra, J. Feldman, and D. Rubenstein. Growth codes: maximizing sensor network data persistence. *SIGCOMM Comput. Commun. Rev.*, 36(4):255–266, Aug. 2006.

[47] F. Kargl, P. Papadimitratos, L. Buttyan, M. Müter, E. Schoch, B. Wiedersheim, T.-V. Thong, G. Calandriello, A. Held, A. Kung, and J.-P. Hubaux. Secure vehicular communication systems: implementation, performance, and research challenges. *Communications Magazine*, 46(11):110–118, 2008.

[48] T. H.-J. 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)*, page 51, New York, NY, USA, 2010. ACM Press.

[49] A. Lazarevic, V. Kumar, and J. Srivastava. Intrusion detection: A survey. In V. Kumar, J. Srivastava, and A. Lazarevic, editors, *Managing Cyber Threats*, volume 5 of *Massive Computing*, pages 19–78. Springer US, 2005.

[50] 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, Oct. 2004.

[51] 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)*, pages 57–66, New York, NY, USA, 2006. ACM Press.

[52] T. Leinmüller, R. K. Schmidt, and A. Held. Cooperative Position Verification-Defending Against Roadside Attackers 2.0. In *Proceedings of 17th ITS World Congress*, pages 1–8. ITS America, 2010.

[53] T. Leinmüller, R. K. Schmidt, E. Schoch, A. Held, and G. Schäfer. Modeling roadside attacker behavior in VANETs. In *2008 IEEE GLOBECOM Workshops*, pages 1–10. IEEE, 2008.

[54] T. Leinmüller, E. Schoch, F. Kargl, and C. Maihöfer. Decentralized position verification in geographic ad hoc routing. *Security and Communication Networks*, 3(4):289–302, 2008.

[55] 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)*, pages 84–91. IEEE, Jan. 2007.

[56] X. Lin, R. Lu, C. Zhang, H. Zhu, P.-H. Ho, and X. Shen. Security in vehicular ad hoc networks. *Communications Magazine*, 46(4):88–95, 2008.

[57] B. Liu, J. T. Chiang, and Y.-c. Hu. Limits on Revocation in VANETs. Pre-proceedings of the 8th International Conference on Applied Cryptography and Network Security (ACNS 2010), 2010.

[58] N.-W. Lo and H.-C. Tsai. Illusion Attack on VANET Applications - A Message Plausibility Problem. In *2007 IEEE Globecom Workshops*, pages 1–8. IEEE, Nov. 2007.

[59] S. Marti, T. J. 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, pages 255–265, New York, NY, USA, 2000. ACM Press.

[60] M. McGurrin. Vehicle information exchange needs for mobility applications exchange. Technical report, United States Department of Transportation, February 2012.

[61] N. I. of Standards and Technology. Digital signature standard. Technical Report FIPS PUB 186-4, Federal Information Processing Standards, july 2013.

[62] E. Palomar, J. M. de Fuentes, A. I. Gonzlez-Tablas, and A. Alcaide. Hindering false event dissemination in VANETs with proof-of-work mechanisms. *Transportation Research Part C: Emerging Technologies*, 23:85–97, Aug. 2012.

[63] P. Papadimitratos, L. Buttyan, T. Holczer, E. Schoch, J. Freudiger, M. Raya, Z. Ma, F. Kargl, A. Kung, and J.-P. Hubaux. Secure vehicular communication systems: design and architecture. *Communications Magazine*, 46(11):100–109, 2008.

[64] P. Papadimitratos, G. Mezzour, and J.-P. Hubaux. Certificate revocation list distribution in vehicular communication systems. In V. K. Sadekar, P. Santi, Y.-C. Hu, and M. Mauve, editors, *Vehicular Ad Hoc Networks*, pages 86–87, New York, NY, USA, 2008. ACM Press.

[65] J. Petit, M. Feiri, and F. Kargl. Spoofed data detection in vanets using dynamic thresholds. In *Vehicular Networking Conference (VNC) 2009*, pages 25 –32. IEEE, nov. 2011.

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

[67] J. Petit, F. Schaub, M. Feiri, and F. Kargl. Pseudonym schemes in vehicular networks: A survey. *Communications Surveys Tutorials, IEEE*, 17(1):228–255, Firstquarter 2015.

[68] J. Petit and S. Shladover. Potential cyberattacks on automated vehicles. *Intelligent Transportation Systems, IEEE Transactions on*, 16(2):546–556, April 2015.

[69] 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, pages 83–92, New York, NY, USA, 2012. ACM.

[70] J. R. Quinlan. *C4.5: programs for machine learning*, volume 1. Morgan Kaufmann, 1993.

[71] D. B. Rawat, B. B. Bista, G. Yan, and M. C. 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*, pages 146–151. IEEE, June 2011.

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

[73] M. Raya, P. Papadimitratos, I. Aad, D. Jungels, and J.-P. Hubaux. Eviction of misbehaving and faulty nodes in vehicular networks. *Journal on Selected Areas in Communications*, 25(8):1557 –1568, oct. 2007.

[74] M. Raya, P. Papadimitratos, V. D. Gligor, and J.-P. Hubaux. On Data-Centric Trust Establishment in Ephemeral Ad Hoc Networks. In *2008 IEEE INFOCOM - The 27th Conference on Computer Communications*, pages 1238–1246. IEEE, Apr. 2008.

[75] M. Raya, P. Papadimitratos, and J.-P. Hubaux. Securing vehicular communications. *Wireless Communications Magazine*, 13(5):8–15, Oct. 2006.

[76] M. Raya, R. Shokri, and J.-P. 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, pages 75–80, New York, NY, USA, 2010. ACM Press.

[77] Z. Ren, W. Li, and Q. Yang. Location verification for VANETs routing. In *Proceedings of the International Conference on Wireless and Mobile Computing, Networking and Communications (WIMOB)*, pages 141–146. IEEE, 2009.

[78] 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)*, pages 827–834. IEEE, Oct. 2011.

[79] S. Ruj, M. A. 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)*, pages 1–5. IEEE, Sept. 2011.

[80] F. Sabahi and A. Movaghar. Intrusion detection: A survey. In *3rd International Conference on Systems and Networks Communications (ICSNC)*, pages 23–26. IEEE, 2008.

[81] 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)*, pages 1–6. IEEE, Apr. 2010.

[82] R. K. 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)*, pages 1–8. IEEE, 2008.

[83] E. Schoch and F. Kargl. On the Efficiency of Secure Beaconing in VANETs. In *Proceedings of the third ACM conference on Wireless network security*, WiSec '10, pages 111–116, New York, NY, USA, 2010. ACM Press.

[84] E. Schoch, F. Kargl, and M. Weber. Communication patterns in VANETs. *Communications Magazine*, 46(11):119–125, 2008.

[85] G. Shafer. *A mathematical theory of evidence*. Princeton university press Princeton, 1976.

[86] H. Stübing, J. Firl, and S. A. 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)*, pages 17–24. IEEE, Nov. 2011.

[87] H. Stübing, A. Jaeger, N. Bißmeyer, C. Schmidt, and S. A. Huss. Verifying mobility data under privacy considerations in Car-to-X communication. In *Proceedings of 17th ITS World Congress*, pages 1–12. ITS America, 2010.

[88] A. Studer, F. Bai, B. Bellur, and A. Perrig. Flexible, extensible, and efficient VANET authentication. *Journal of Communications and Networks*, 11(6):574–588, Dec. 2009.

[89] K. L. Thng, B. S. Yeo, and Y. H. Chew. Performance study on the effects of cell-breathing in wcdma. In *2nd International Symposium on Wireless Communication Systems*, pages 44–49. IEEE, 2005.

[90] C. Troncoso, G. Danezis, E. Kosta, J. Balasch, and B. Preneel. PriPAYD: privacy-friendly pay-as-you-drive insurance. *Transactions on Dependable and Secure Computing*, 8(5):742–755, 2011.

[91] K. Wang and S. J. Stolfo. Anomalous payload-based network intrusion detection. In E. Jonsson, A. Valdes, and M. Almgren, editors, *Recent Advances in Intrusion Detection*, number 3224 in Lecture Notes in Computer Science, pages 203–222. Springer Berlin Heidelberg, Jan. 2004.

[92] 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, pages 176–183, Piscataway, NJ, USA, 2010. IEEE Press.

[93] 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)*, pages 1–8, New York, NY, USA, 2006. ACM Press.

[94] G. Yan, B. B. Bista, D. B. Rawat, and E. F. Shaner. General Active Position Detectors Protect VANET Security. In *Proceedings of the 2011 International Conference on Broadband and Wireless Computing, Communication and Applications*, pages 11–17. IEEE, Oct. 2011.

[95] G. Yan, S. Olariu, and M. C. Weigle. Providing VANET Security Through Active Position Detection. *Computer Communications*, 31(12):2883–2897, July 2008.

[96] 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 Y. Ishikawa, J. Li, W. Wang, R. Zhang, and W. Zhang, editors, *Lecture Notes in Computer Science: Asia-Pacific Web Conference 2013*, volume 7808, pages 583–594. Springer, 2013.

[97] T. Zhou, R. R. Choudhury, P. Ning, and K. Chakrabarty. Privacy-preserving detection of sybil attacks in vehicular ad hoc networks. In *Fourth Annual International Conference on Mobile and Ubiquitous Systems: Networking Services*, pages 1 –8. IEEE, aug 2007.

[98] T. Zhou, R. R. Choudhury, P. Ning, and K. Chakrabarty. P2DAP — Sybil Attacks Detection in Vehicular Ad Hoc Networks. *Journal on Selected Areas in Communications*, 29(3):582–594, Mar. 2011.

[99] Y. Zhou, Y. Fang, and Y. Zhang. Securing wireless sensor networks: a survey. *Communications Surveys Tutorials*, 10(3):6–28, 2008.

[100] 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*, page 106, New York, NY, USA, 2009. ACM Press.