Eviction of Misbehaving and Faulty Nodes in Vehicular Networks

M. Raya†, P. Papadimitratos†, I. Aad‡, D. Jungels, and J.-P. Hubaux†

† LCA Lab, EPFL, Switzerland
‡ Future Networking Lab, DoCoMo Euro-Labs, Germany
firstname.lastname@epfl.ch, aad@docomolab-euro.com, daniel.jungels@a3.epfl.ch

Abstract

Vehicular Networks (VNs) are emerging, among civilian applications, as a convincing instantiation of the mobile networking technology. However, security is a critical factor and a significant challenge to be met. Misbehaving or faulty network nodes have to be detected and prevented from disrupting network operation, a problem particularly hard to address in the life-critical VN environment. Existing networks rely mainly on node certificate revocation for attacker eviction, but the lack of an omnipresent infrastructure in VNs may unacceptably delay the retrieval of the most recent and relevant revocation information; this will especially be the case in the early deployment stages of such a highly volatile and large-scale system. In this paper, we address this specific problem. We propose protocols, as components of a framework, for the identification and local containment of misbehaving or faulty nodes and then for their eviction from the system. We tailor our design to the VN characteristics and analyze our system. Our results show that the distributed approach to contain nodes and contribute to their eviction is efficiently feasible and achieves a sufficient level of robustness.

This work was done during I. Aad’s and D. Jungels’ stay at EPFL.
I. INTRODUCTION

Recent research initiatives supported by governments and car manufacturers seek to enhance the safety and efficiency of transportation systems. Vehicular networks lie at the core of these efforts. Vehicular network nodes, that is, vehicles and Road-Side infrastructure Units (RSUs) will be equipped with sensing, processing, and wireless communication modules. Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communication will enable safety applications that provide warnings about accidents, traffic conditions (e.g., congestion, emergency braking) and other events.

Integrating security mechanisms into the VNs is critical for their deployment: their rich functionality and services can be otherwise abused, jeopardizing the safety of vehicles, drivers, and passengers, as well as the efficiency of the transportation system. A number of research contributions analyze vulnerabilities [6], [21], outline architectural components, requirements, and design principles [20], and propose specific mechanisms [3], [11], [15], [23], [28].

The presence of an authority, which we denote as the Certification Authority (CA), is implied or mandated in practically all the research efforts concerned with securing VNs. Rigid identity and credential management processes for vehicles and drivers have long been in place; accountability and attribution of liability will continue to be crucial; and access control mechanisms will be necessary. Without the appropriate certificate and cryptographic keys, nodes are essentially unable to participate in the network operation. Nevertheless, the possession of a certificate does not guarantee that its holder will provide correct information: a node can simply inject faulty data (e.g., alerts, warnings, coordinates) while complying with the implemented protocols. Safeguarding the system against such faulty or compromised nodes is crucial for its robustness. Hence the need for the eviction of misbehaving nodes. A typical approach for achieving this is the revocation of their certificates; once this is done, messages from these nodes will be ignored.

Timely access to revocation information is a particularly hard problem in VNs. The road-side infrastructure can act as the gateway of the CA to the network, distributing the latest Certificate
Revocation Lists (CRLs) [13]. The lack of an omnipresent road-side infrastructure, especially in the early deployment stages, and the huge scale of the VNs are obstacles to the application of traditional certificate revocation schemes. Moreover, unless a node is revoked for administrative reasons (e.g., the vehicle owner did not renew its registration), how can the authority obtain and validate sufficient evidence that a node is faulty or compromised? Thus, an additional challenge is how non-misbehaving nodes can be protected until they obtain the revocation information regarding misbehaving nodes.

Our contributions in this paper address these problems. We propose the combination of (i) infrastructure-based revocation protocols, the Revocation of the Trusted Component (RTC) and Revocation using Compressed Certificate Revocation Lists (RC$^2$RL), (ii) a Misbehavior Detection System (MDS) enabling the neighbors of a misbehaving or faulty node to detect its deviation from normal behavior, and initiate (iii) a Local Eviction of Attackers by Voting Evaluators (LEAVE) protocol to safeguard the system operation, until the attacker is revoked by the CA, partially or fully based on the evidence LEAVE provides.

We emphasize however that no group of nodes has the power to revoke another node. The CA is the sole entity that has the right to initiate a revocation protocol. This design choice ensures resilience to collusion attacks, retains accountability, and yet equips nodes with a rapid reaction and self-protection tool. Indeed, our performance evaluation results show that a high percentage of the attacker's neighbors can be alerted of its misbehavior, despite the very short contact time between the protocol participants.

This paper is organized as follows. We first describe the system and adversary models in Sec. II and then provide an overview of our scheme in Sec. III. Next, we present its components in further detail in Sections IV-VI. We evaluate our scheme in Sec. VII. Then, we survey related work in Sec. VIII. Finally, we conclude in Sec. IX.
II. System Model

Drawing from the analogy with existing administrative processes and automotive authorities (e.g., city or state transit authorities), a large number of CAs will exist. Each of them is responsible for the identity management of all vehicles registered in its region (national territory, district, county, etc.). Vehicles are registered with exactly one CA. Each node has a unique identity $V$ and a pair of private and public cryptographic keys, $PrK_V$ and $PuK_V$, respectively, and is equipped with a certificate $Cert_{CA}\{V, PuK_V\}$ issued by the CA. The vehicle may have several keys for privacy reasons [23] but this does not affect the operation of the proposed mechanisms (because revocation and eviction apply actually to the keys of a vehicle) and is out of scope of this paper.

Messages are transmitted periodically, e.g., every 0.3 s for safety messages, or triggered by in-vehicle or network events. Most of the traffic is broadcasted to limited regions of the network; the broadcast regions are determined by the corresponding applications. All safety-related messages include the time and geographical coordinates (obtained by a positioning service, such as the widely available GPS) of the sender, in addition to other application-specific information. In addition, each message is signed and accompanied by the sender’s certificate.

Safety messages that need to propagate across multiple hops (and perhaps have the originator’s signature, coordinates and time intact as they propagate) are signed and have the coordinates and timestamp of the last relaying node to ensure the freshness of the information and to limit the propagation of false information. A received safety message is discarded if the difference between its timestamp and the timestamp of the receiver is larger than a system-specific constant that accounts for the maximum clock drift and one-hop transmission, propagation and processing delays. Moreover, a message is discarded (at a receiver) if the coordinates of its sender/relay, as reported in the message, indicate that the receiver is outside the sender’s maximum nominal wireless communication range (accounting for location information inaccuracies). This validation is applied only on a per-hop manner.
At the data link layer, the Dedicated Short Range Communications (DSRC) protocol, currently being standardized as IEEE 802.11p [2], provides transmission ranges of typically 300 to 1000 m, with data rates in the 6-27 Mbps range. In this paper, we assume that 802.11p is used, unless noted otherwise. Beyond DSRC, vehicular networks can leverage on other wireless communication technologies, such as the (licensed-frequency) existing cellular networks, broadband wireless (e.g., WiMax), or low-speed radio broadcast systems used today for traffic information.

We denote a subset of the network nodes as the infrastructure, comprising the RSUs (i.e., short-range DSRC base stations) and mobile units. The latter include public safety vehicles (e.g., highway assistance and fire-fighting vehicles), police vehicles, aerial vehicles (e.g., police helicopters), and public transport vehicles (e.g., buses, trams). These nodes can be used, for example, to disseminate CRLs in our context. Infrastructure nodes serve as the gateway of the CA to/from the VN; the connection of the CA to the static infrastructure nodes is over wireline secure links. We note however that accessibility of the CA from the VN is not assumed to be guaranteed at all times.

Many vehicles are already equipped with hardware and firmware components, such as speed limiters, tachographs, and event data recorders (EDRs), which are considered critical by manufacturers and legislators. We assume that nodes are equipped with a Trusted Component (TC), i.e., tamper-resistant hardware and firmware. The role of the TC is two-fold: (i) it stores all cryptographic material and prevents its exposure to the on-board computer; (ii) it performs all cryptographic operations.

A. Adversary Model

We denote as an adversary or attacker any node implementing VN protocols that deviate from the legitimate operation. Nodes can also be faulty due to failures of their equipment. A detailed discussion of adversary and fault models is given in [20]. Any of these attacks or faults, or combinations thereof, can affect the VN-enabled applications. We also refer to adversaries as
misbehaving nodes. As our proposed mechanisms apply to both misbehaving and faulty nodes, we will use both terms interchangeably in the remainder of this paper without losing the generality of the solutions.

In addition, the information-oriented operation of VNs, with their diverse data types, makes the false information dissemination a very effective attack, compared to deviations from the networking protocols. In fact, it would suffice for an adversary to manipulate the sensory inputs rather than to compromise the protocol stack and the computing platform. It is also possible that an attacker controls incoming communication, i.e., selectively erasing messages received by its on-board platform.

We emphasize that we are concerned with misbehaving nodes equipped with valid credentials, because they can then effectively abuse the system. An essential assumption we make is the existence of an honest majority in the attacker’s neighborhood (defined in Sec. VI-B). As we will show later, this allows vehicles to rely on the honest neighbors in order to evict attackers. This assumption (also elaborated in [12], [20]), although it appears limiting, is reasonable if we consider the existing transportation systems where the actual percentage of attackers is very low.

III. SCHEME OVERVIEW

Our scheme consists of the following basic components: (i) the centralized revocation of a node by the CA, (ii) the local detection of misbehavior, performed individually by each node and (iii) a distributed, localized protocol for the eviction of the attacker by its neighboring nodes. The scheme with its components is illustrated in Fig. 1.

We propose two methods for misbehaving node revocation, initiated by the CA. The first one, **RTC** (Revocation of the TC, described in Sec. IV-A), leverages on the presence of a TC unit on-board the vehicle. The CA determines that a vehicle $V$ must be revoked and, with the help of the road-side infrastructure, initiates a two-party end-to-end protocol with $TC_V$, the trusted
component of $V$. The CA instructs the TC to erase all cryptographic material (e.g., keys) it stores and halt its operation upon completion of the protocol. Essentially, this protocol “kills” the TC, depriving the misbehaving node from its cryptographic key, and it ensures that all its messages are ignored by all other correct nodes.

As the misbehaving node is targeted directly, RTC is not robust against a sophisticated adversary that controls the communication link between the CA and the TC. If the CA fails in executing RTC (by lacking an acknowledgment), it will revert to the distribution of the revocation information, namely, a CRL, to the VN. Thus, the CA invalidates credentials before the end of their lifetime. But the size of CRLs will grow with the size of the VN and hence is not scalable. To adapt this approach to the VN scale, we propose the $RC^2RL$ ($Revocation using Compressed Certificate Revocation Lists$) protocol, with $C^2RL$s shorter than traditional CRLs by means of Bloom filter compression. The protocol is detailed in Sec. IV-B.

The timely and efficient distribution of revocation information across the VN is the primary means of revoking misbehaving nodes. However, to design a robust and efficient system capable of progressively isolating misbehaving nodes before this information becomes available, we propose the use of a localized $MDS$ ($Misbehavior Detection System$) and the $LEAVE$ ($Local Eviction of Attackers by Voting Evaluators$) protocol.

$MDS$, discussed further in Sec. V, is an essential enabler of LEAVE. Each node uses its own sensory inputs (including time and location), messages received from its neighbors (assuming an honest majority), and a set of evaluation rules, to classify safety messages received from a given node as faulty or correct. Messages that are outdated (aged), received beyond their expected area of propagation, or contradictory to the node’s own state\footnote{For example, a traffic jam message received when the node’s velocity in the allegedly jammed area is well above the velocity that corresponds to the onset of a traffic jam.} are considered false. Their senders, as long as they are neighbors of the node running $MDS$, are also tagged as misbehaving. Then, their identity is passed to LEAVE.
The main principle of LEAVE, detailed in Sec. VI, is simple: the neighbors of the misbehaving vehicle temporarily “evict” it. In contrast to RTC and RC\textsuperscript{2}RL, LEAVE is not a revocation protocol, but rather a collective warning system against misbehaving nodes. Upon detecting an attacker, vehicles broadcast warning messages to all vehicles in range, so that the sharing of information improves the effectiveness of the stand-alone detection systems. Moreover, such warnings can be invaluable when vehicles receive them before being able to observe the misbehaving node themselves. We clarify that the notion of neighborhood is different for MDS and LEAVE. In the first case, it includes all one-hop neighbors of the vehicle running the MDS. LEAVE, as further detailed in Sec. VI-B, elects a subset of this neighborhood; this subset depends on both the vehicle running the MDS and the attacker.

The eviction of an attacker by its neighbors is temporally limited to the duration of contact between the attacker and its neighbors running LEAVE. But once enough evidence against the attacker is gathered, the CA can initiate one of the previously described revocation protocols. Recall that the CA is the only system entity entitled to revoke keys (given all the related administrative responsibilities and costs). In this paper, we do not consider the CA decision process for node revocation, as a number of legal and policy aspects are involved. In addition, the reasons for revocation are largely orthogonal to the operation itself and can include administrative procedures (e.g., change of registration domain), cryptographic material compromise (e.g., a private key was detectably disclosed) or, as mentioned above, node misbehavior for which the CA obtains sufficient evidence.

IV. REVOCATION PROTOCOLS

A. Revocation of the Trusted Component (RTC)

When the CA decides to revoke a vehicle $V$, it first uses RTC (Fig. 2): The CA generates a revocation message for $TC_{V}$; this message contains $V$’s unique identity and a timestamp $T$, encrypted with $V$’s public key $PuK_{V}$, and signed by the CA. Thus, $TC_{V}$ and the RSUs that
forward the message can verify its authenticity and freshness. The message format is:

\[ CA \rightarrow TC_V : E_{PuK_V}(V|T), \text{Sig}_{CA}[E_{PuK_V}(V|T)] \]

where \( E_{PuK_V}() \) denotes encryption with public key \( PuK_V \).

There are several options for channeling this message to the targeted TC. The first choice would be to route it to the RSU closest to the concerned vehicle, if its location is known to the CA. Otherwise the CA defines a paging area, consisting of several RSUs in the region of the vehicle’s most recent locations (trajectory extrapolation based on the vehicle’s expected speed and acceleration can be useful in determining the paging area). If all else fails, the CA can use other distribution media mentioned in Section II, such as low-speed radio broadcast.

When \( TC_V \) receives the RTC message, it immediately erases the cryptographic key and stops signing VN messages. It sends back a timestamped and signed acknowledgment, as soon as it comes within range of a RSU: \( TC_V \xrightarrow{RSU} CA : \text{ACK}, T, \text{Sig}_{PrK_V}[\text{ACK}|T] \).

If the vehicle \( V \) is an attacker capable of blocking messages destined to its TC, the CA will receive no acknowledgement and thus will detect the failure of RTC. It will then revert to the RC\(^2\)RL protocol discussed in the next subsection.

### B. RC\(^2\)RL (Revocation using Compressed Certificate Revocation Lists)

As CRLs contain very little redundancy, they cannot be efficiently compressed using normal lossless methods. We therefore use Bloom Filters [5], a special form of lossy compression, to generate C\(^2\)RLs (Compressed CRLs) that the CA signs and broadcasts using one of the previously mentioned distribution methods. Bloom filters provide a probabilistic data structure used to test whether an element is a member of a set. They are characterized by a configurable rate of false positives and no false negatives. This ensures that the CA can efficiently revoke all targeted nodes while keeping false revocations within acceptable error margins. A more detailed explanation of Bloom filters and their specific application to revocation in VNs can be found in Appendix I.
V. MISBEHAVIOR DETECTION SYSTEM

As explained in Sec. II-A, it may be more beneficial for the adversary in VNs to disrupt the data transferred by protocols rather than the protocols themselves. Hence, the Misbehavior Detection System (MDS) should rely not only on the protocol-specific actions of nodes but also on the data these nodes provide. Similarly to Intrusion Detection Systems (IDS) [19], we can distinguish between two types of misbehavior in VNs:

1) **Known misbehaviors** that can be identified by monitoring specific parameters of node or network behavior. These misbehaviors include typical intrusion detection metrics (e.g., in routing protocols), known viruses and worms, and MAC-layer anomalies.

2) **Data anomalies** that do not follow any known pattern. This is often the case when the adversary modifies safety messages according to its specific needs.

Known misbehaviors can be detected using their signatures, in IDS terms. The comparison to a defined threshold is a simple, yet efficient, detection technique for many misbehaviors; the choice of the threshold depends on the misbehavior and its context.

Standard IDS techniques detect data anomalies by monitoring specific metrics, comparing the actual metric values to expected values, and by thresholding the deviation to detect attacks. The problem with applying these approaches to VNs is that expected values are not known in advance and do not always fit a given model. For example, the traffic congestion varies considerably depending on the road and the time of the day. A more adaptive approach consists in comparing data to itself (i.e., comparing the behavior of each node to the average behavior of the other nodes, including the node running the MDS) or building data models on the fly, which is a more general approach. To do the latter, we use entropy, a typical measure of information\(^2\).

Assuming \( n \) reporting nodes (the sample size \( n \) depends on the application), the entropy is given by: \( H = - \sum_{i=1}^{n} p_i \log_2 p_i \) where \( p_i \) is the probability that node \( i \) is an attacker. The computation of \( p_i \) depends on the application; we will provide a detailed example in the next subsection.

\(^2\)Feinstein et al. [10] have also proposed using entropy for the detection of Denial-of-Service attacks using packets headers.
If all nodes are well-behaving, then they can be attackers with the same probability. We call
the corresponding probability density function $P_0$. To reduce false positives, the MDS should
tolerate mild faults up to a given error margin $\epsilon$. Let $P_\epsilon$ be the corresponding probability density
function. The MDS detects an attack (i.e., an anomaly) in the system if the corresponding attacker
probability density function has the following property:

$$\frac{D(P\|P_0)}{D(P_\epsilon\|P_0)} > R$$

(1)

where $D(P\|Q)$ is the Kullback-Leibler distance [17] between the two probability density
functions $P$ and $Q$; $R$ is the detection threshold for anomalous distance ratios.

As entropy alone only reflects the state of the system without identifying the attackers, we
use an outlier detection algorithm [24] to single out the attackers. Among the possible options,
we use the K-means clustering algorithm [14] that iteratively and efficiently converges to the
number of clusters that minimizes the sum of distances of all points to the corresponding cluster
centroids. The use of a clustering algorithm is justified only if all nodes report similar information
with varying deviations. If nodes report contradicting data, a simple voting mechanism can be
used to decide which nodes to believe; this takes advantage of the honest majority. Based on
the above, the operation of the MDS can be summarized as shown in Fig. 3.

A VN application determines the corresponding known misbehaviors, algorithm for computing
the $p_i$ values, and reference probability density functions $P_0$ and $P_\epsilon$. We describe in detail an
example application of the proposed MDS below.

It should be noted that the MDS detects only nodes that are in its current neighborhood, based
on their locations and timestamps. This limits the load of attacker detection to the attacker’s
neighbors, thus minimizing overhead. This way, the MDS does not distinguish between data
originators and data relays. In fact, a relay node that propagates false data constitutes a vulnerable
point and needs to be contained.
A. Example Application of the MDS

In a typical VN application, a vehicle stopped on the roadside emits warnings destined to all vehicles within a 1 km range. The purpose of these warnings is to alert other vehicles of an event, such as an accident, that requires these vehicles to slow down. As the typical range of DSRC is 300 m, intermediate vehicles need to forward the warnings over several hops after verifying and signing them (to prevent the uncontrolled propagation of false information). We investigate how the MDS running on a vehicle would react to the presence of attackers.

We should note here that the observing vehicle does not change its position significantly during the execution time of the MDS because all warnings are transmitted over the high-speed DSRC and all computations are based on efficient algorithms and are done locally.

We assume that the attackers attempt, in their warnings, to enlarge the actual distance from the event, e.g., to cause an accident (because vehicles will not brake in time). Reporting smaller distances in this case would not accomplish the desired attack (because vehicles will brake anyway if the distance is critically small). We consider the case where the observing vehicle is several hops away from the event but within 1 km.

First, the observing vehicle computes the \( p_i \) values corresponding to reporting vehicles as follows. Assuming that the observing vehicle is at location \((x_o, y_o)\) and that vehicle \( i \) reports that the event/accident is at location \((x_e, y_e)\), the resulting computed distance from the event is \( d_i = d((x_e, y_e), (x_o, y_o)) \) where \( d \) denotes a distance function (e.g., Euclidean distance). Let \( q_i \) be the ratio of \( d_i \) to the average of the lowest 50% of the reported distances (assuming that attackers only enlarge distances and there is an honest majority). Finally, \( p_i = \frac{q_i}{\sum_{i=1}^{n} q_i} \) is the probability that vehicle \( i \) is an attacker. The MDS then applies the detection rule in Equation 1.

The reference probability density functions \( P_0 \) and \( P_\epsilon \) can be computed based on the sample size \( n \) and the tolerable error margin \( \epsilon \). In this example, \( n \) is equal to the number of reports.

---

3Assuming \( n \) neighbors, the approximation of entropy is upper-bounded by \( O(n^{o(1)\log n}) \) [4]; the complexity of K-means is \( O(n) \) [14].
(about the event) received from distinct neighbors.

Fig. 4(a) shows, in the case of 100 nodes and an error margin of 10%, the performance of the MDS corresponding to different percentages of attackers and percentages of distance enlargement (or excess) that they report. The detection threshold $R$ (i.e., deviation ratio) is set to 2 in this example. We can see that the MDS can detect, for a 30% excess, attacks resulting from 5% to 91% of attackers. Larger attacker percentages cannot be detected because they strongly influence the reference values. Fig. 4(b) shows the percentage of attackers detected by K-means clustering. We can notice that the larger the percentage of excess (i.e., distance enlargement), the easier it is to detect attackers. This is rather intuitive and acceptable because mild abuse can in fact be a small detection system error due to the highly dynamic environment.

VI. THE LEAVE PROTOCOL

As mentioned in Section III, being warned of the misbehaving nodes allows the observing vehicle to ignore any messages sent by these nodes. Warnings can be triggered by the standalone MDSs running on each vehicle. This is the key concept behind LEAVE, shown in Fig. 5. More precisely, vehicles that detect an attacker will broadcast warning messages to all vehicles in range. The latter can use this information as input to their respective MDSs. In this paper, we consider the case of vehicles that receive warning messages before they are able to make any observations of the attacker and that, as a result, totally rely on these messages. The final step is to report the attackers or faulty nodes to the CA as soon as possible (i.e., when in reach of a base station or mobile unit as defined in Sec. II).

A. Neighbor Warning System for LEAVE

A warning system that can be useful in LEAVE should rely on the collective information gathered from a vehicle’s neighborhood. As all vehicles can be attackers with the same probability, the exchanged warning messages may contain correct or wrong accusations. Given the

\footnote{We can notice that the assumption of an honest majority is not necessary in this particular example.}
limited amount of evidence available to vehicles receiving this information, they have to rely on
the assumption of honest majority and crosscheck all the received accusations.

In this paper, we use a simple algorithm, explained in Appendix II-A, for summing accusations
with an additional feature inspired by [9]: An accusation issued by a node has a lower weight
when this node is already accused by other participants. For a given vehicle, if the sum of
weighted accusations (the \textit{eviction quotient}) against it exceeds a defined threshold, it is locally
evicted by LEAVE. More precisely, warning messages are transformed into \textit{disregard} messages
that instruct all the neighbors of the attacker to ignore its messages.

It should be stressed here that this algorithm is only an example and other accusation ag-
gregation systems can be devised. We chose this rather simple system because it requires no
setup overhead, like incentive systems [30], nor long observation periods, like reputation systems
[7]. In addition, it involves no interactive mechanisms, thus preventing our system from being
dependent on specific participants. As stated earlier, the only requirement for the proposed
neighbor warning system is the existence of an honest majority.

The difference between warning and disregard messages is that a specific number of \textit{supporting
signatures} is included by the sender in the disregard message. This increases the credibility of
the message, under the assumption of an honest majority, while maximizing channel efficiency
by message aggregation [22].

A vehicle that has accumulated enough accusations against an attacker to reach the \textit{eviction
threshold} (its computation is detailed in Appendix II-A) and disregard messages from it is called
hereafter a \textit{warned vehicle}. An \textit{initially warned vehicle} is a warned vehicle that disregards
all bogus messages from the attacker. This can happen if the vehicle has already reached the
warned state before receiving the first message from the attacker (because it has received enough
accusations to reach the eviction threshold).

It should be noted that, although LEAVE relies on accusations by nodes, it is highly resilient
to attackers as shown in Appendix II-B.
B. Definition of the Attacker’s Neighborhood

The neighbor warning system, introduced in the previous section, relies on the neighbors of a suspected vehicle (suspect) to accuse it in case of misbehavior and warn other vehicles. Once these vehicles have enough information about the suspect, they can evaluate whether it is misbehaving; we refer to them as evaluators. Hence, it is important to define the suspect’s neighborhood $N$. New vehicles coming into the communication range of a suspected vehicle will use the information provided by their neighbors to detect misbehavior. It is essential here to avoid the hidden vehicle problem whereby a vehicle that is not in the suspect’s range, and hence cannot evaluate it, is considered when making a decision (Fig. 6). Therefore, the neighborhood $N$ of a suspected vehicle also depends on the vehicle evaluating it. We define $N(s, e)$, where $s$ and $e$ refer to suspect and evaluator, respectively, as the intersection of the coverage areas (defined by the transmission range) of both the suspected vehicle and the vehicle evaluating it. In this way, we make sure that all vehicles in $N$ can hear the suspect, and thus potentially accuse it, and at the same time be able to report their accusations to the evaluators.

It should be noted that an evaluator vehicle can select the elements of $N$ because all vehicles broadcast their position information. The suspect’s position is reported in the warning messages received by the evaluator.

VII. Performance Evaluation

In this section, we evaluate the performance of LEAVE under stringent VN conditions. The following results show that the eviction of misbehaving nodes can be done efficiently. A detailed performance analysis of the Bloom filters can be found in Appendix I-A.

As LEAVE relies on the ad hoc operation of vehicles within short time delays, we simulate it using ns-2 with the MAC-layer parameters of IEEE 802.11p. We consider three different parameters in our evaluations. The first one is the traffic model: we used a freeway (FW),
a city (West University, or WU), and a mixed (freeway/city) (Afton Oak, or AO) scenarios. The second and third factors are the density of vehicles and their average speed, respectively. The presented results are the average of 50 simulation runs. As the MDS runs locally on each vehicle, its computational delay is small and hence not critical for the system operation; in the simulations, we do not include this delay. There is one adversary detected by the MDS, but the same results apply to multiple adversaries as they are independent (LEAVE runs separately for each adversary).

A. Vehicle Density

In the WU and AO scenarios, we can see that for a very low vehicle density the percentage of initially warned vehicles (Section VI-A) is low (Fig. 7(a)). In this case it is not possible to pass information from one vehicle to another in a reliable way and to warn other vehicles beforehand. However, this value grows with increasing density and stabilizes between 80% and 90%. For those vehicles not initially in the warned state, the average time to reach the threshold in order to disregard messages stabilizes at a value lower than 2 s (Fig. 7(b)). For the freeway scenario, there is a slight decrease in performance at very high densities. This can be explained by the fact that the number of packet collisions increases when the density increases. Because of the hidden node problem, even though we are not yet at channel capacity, the number of colliding packets increases and creates some loss. As in this scenario speeds are higher (and thus contacts shorter), this may result in the fact that some accusations are not received by every neighbor.

From this we can conclude that even with a relatively low density of VN-enabled vehicles, which will be the case with a low market penetration at the beginning of VN deployment, LEAVE is still able to accumulate enough information to perform successfully within the delay bounds imposed by short contact times (e.g., assuming a transmission range of 300 m and two vehicles

\(^5\)WU and AO are realistic scenarios taken from [25]. The framework used for developing the simulations in this section is available at [1].
moving in opposite directions on the same highway, each at an average speed of 100 km/h, their contact time is merely around 11 s).

B. Average Speed

We evaluate the same metrics for different average speeds in the three scenarios (Fig. 7(c) and 7(d)). In the urban areas (AO and WU), we can see that higher speeds give better results: again, because more participants can be contacted in the same time interval, it is possible to accumulate more accusations against the attacker and to warn other vehicles (in these two cases, the maximum average speed is 90 km/h, higher values are rare). In the freeway scenario, the average speed is much higher, and performance decreases slightly for very high speeds: this can be explained by the fact that the contact time becomes very short (approaching the same order of magnitude as accusation message sending). In this case, some messages may not be received, thus resulting in a slightly higher time to reach the necessary threshold. Still, we can see that LEAVE can execute within acceptable delays and cover a considerable percentage of concerned vehicles.

C. Effect of Warning Rebroadcast Interval

If a vehicle continues to receive bogus messages from the attacker, it does not send additional warning messages, unless the difference between the vehicle’s current time and the timestamp of its last sent warning message (against the attacker) is larger than the Warning Rebroadcast Interval (WRI) parameter. The WRI is used to prevent vehicles from flooding the channel by sending an accusation message every time they receive a bogus message; this prevents a possible DoS attack based on excessive channel load or on the computation overhead due to digitally signing warning messages. However, if a vehicle sends a warning message only once and then stops participating in the warning process against a potential attacker, newly arriving vehicles will not be able to accumulate enough accusations to disregard the attacker’s messages. Hence,
the parameter WRI is a tradeoff between sending too often the same warnings, and quickly informing new neighbors about misbehaving vehicles.

Changing the WRI has only a small effect on the percentage of initially warned nodes in the case of low vehicle density (Fig. 7(c)). In higher density situations we can observe a slight increase of warned vehicles with smaller WRI values when sending accusations more frequently, but this also increases channel load. As the results show, WRI can be kept large without much degradation in the performance of LEAVE, but with a considerable reduction in channel load.

D. Number of Supporting Signatures

The number of supporting signatures (Section VI-A) is a parameter of LEAVE. It may be influenced by the cryptographic system that is used: If signature sizes are large, the number of supporting signatures may have to be kept small in order to maintain a reasonable packet size. But a minimum number of signatures is still required to assure a sufficient level of credibility to disregard messages. Fig. 7(f) shows that the number of warned vehicles changes considerably with small numbers of supporting signatures. But starting from only 4 signatures, there is little change in the results. This can be explained by the fact that, starting from this point, the number of vehicles that broadcast exactly the same supporting signatures (and thus give no new information) increases.

To cope with the abuse of disregard messages, a vehicle should still crosscheck several disregard messages and verify that the total number of supporting signatures received in these messages is larger than the majority of nodes in the neighborhood $N$ (assuming an honest majority). The advantage of relying on small rather than large disregard messages is smaller retransmission overhead if a collision happens on the wireless channel.

VIII. RELATED WORK

Revocation has been considered mostly in the context of the wireline Internet and the design of Public Key Infrastructure (PKI) services [13]. Nevertheless, the design of mechanisms to
disseminate the revocation information across systems similar to VNs has not been considered in the wireline Internet context (for a survey and discussion of tradeoffs see [26], [29]). Due to the network volatility and scale, the overhead of querying a server to obtain timely revocation status, assuming the server is reachable, would be impractically high. For the same reasons, schemes that distribute the load of a server to a set of participating clients [27] (to redundantly forward revocation information) would not be practical for deployment within the VN, but only meaningful behind the fixed infrastructure.

Existing works on VN security [21], [23], [28] propose the use of a PKI and digital signatures but do not provide any mechanisms for certificate revocation, even though it is a required component of any PKI-based solution. In the context of VNs, the IEEE 1609.2 Draft Standard [3] is the only reference on certificate revocation. It proposes the distribution of CRLs and short-lived certificates, but does not elaborate how to achieve this. Short-lived certificates are also proposed in [15], which relies on Merkle tree constructions but does not consider revocation. Short lifetimes are essentially a means of revocation that achieves efficiency but leaves a vulnerability window; such an approach is not appropriate for a life-critical VN environment. Moreover, certificates have to be refreshed frequently because the vulnerability window must be very small in VNs. This creates an overwhelming load both on the CA and the network.

The detection of adversaries (faulty or malicious) in networks, both wired and wireless, is typically done by Intrusion Detection Systems (IDS) [8], [19]. IDSs function according to two models: signature-based attack detection and anomaly detection. The first method is tailored to recognize known attacks; the second method builds a typical behavior model of the system and signals significant deviations from this model as attacks. Most IDSs use packets of various protocols (e.g., TCP, HTTP) to monitor the network; this also applies to wireless networks where IDSs typically monitor routing protocols (e.g., AODV, DSR).

*An exception can be context-specific credentials, allocated, for example, to a vehicle entering a highway segment and “purchasing” access to a service. However, this is orthogonal to the problem we are considering here.
The existing literature on VNs already contains some methods for adversary detection. For example, several threshold-based tests to verify positioning information in VNs are proposed in [18]. In [12], a more general framework for malicious data detection compares the received data to a model of the VN; but the paper provides no details on possible tests.

Instantiating a CA in the context of mobile ad hoc networks was investigated, with the distribution of its functionality to a number of servers by means of threshold cryptography that enhances the robustness and accessibility of the CA [31]. However, this scheme does not consider the problem of revocation, especially in a highly mobile environment like a VN. Instantiation of the CA functionality (or part thereof) by impromptu coalitions of network nodes (e.g., [9], [16]) cannot be applicable in VN systems. Allowing any ad hoc and, in general, small subset of adversarial nodes to maliciously accuse and evict legitimate nodes would be an unacceptable breach of the VN system security where accountability and liability are mandatory.

IX. Conclusion

In this paper, we propose a framework to thwart internal attackers in vehicular networks. The eviction of faulty or attacking nodes is crucial to the robustness of vehicular communication systems. As revocation is the primary means to achieve this, we designed two protocols tailored to the characteristics of the VN environment. To eliminate the vulnerability window due to the latency of the authority to identify those nodes and distribute revocation information, we designed a scheme that can robustly and efficiently achieve isolation of misbehaving and faulty nodes, as well as contribute to their eventual revocation. This is done with the help of a misbehavior detection module and a distributed eviction protocol. These protocols together cover the whole spectrum of VN scenarios. Given the broad scope of the subject tackled in this paper, there is ample space for future work on each of the individual components of our framework. Nonetheless, our results evaluating the instantiation proposed in this paper show that our scheme is practical, and can efficiently and effectively isolate misbehaving and faulty nodes.
REFERENCES


**APPENDIX I**

**Bloom Filters**

A Bloom filter, illustrated in Fig. 8, consists of a $m$-bit vector with all its bits initially set to zeros. An element (a public key in our context) can be included in the filter by (i) hashing it with $k$ independent hash functions that output numbers in the range $1, \ldots, m$, and (ii) setting to 1 the vector bit each hash function points to. It is possible that one bit is set to 1 multiple times due to several element additions. To check if a given element is contained in the filter, the element is hashed and the corresponding filter bits are checked: If at least one of those bits is zero, the element is not contained in the filter. Otherwise, if all necessary $k$ bits are set, the element is included with high probability. The corresponding bit could have been set also due
to multiple additions of other elements. The more elements added, the larger the probability of false positives. In the context of revocation by C²RLs, the nodes validate the certificates included in received messages by checking the Bloom filter. We discuss quantitative aspects of Bloom filters in the next section.

A. Bloom Filter Performance

As mentioned previously, Bloom filters provide a compression tool with configurable compression gain ($c$) and false positives rate ($p_{fp}$). The configurable parameters of the filter are:

- The filter vector size $m$.
- The number of hash functions $k$.

An additional input, however not configurable, is the number of list entries (certificate IDs) $L$, each considered to be $l$ bits long.

A large value of $m$ considerably reduces the false positives rate $p_{fp} = (1 - (1 - 1/m)^{kL})^k \approx (1 - e^{-kL/m})^k$, at the cost of decreasing the compression gain $c = L \times l/m$. The choice of $m$ can be derived from Fig. 9(a), where the number of hash functions $k$ is chosen to be optimal ($ln(2) \times m/n$, when $dp_{fp}/dk = 0$). Taking also into consideration the range of the number of list entries $L$, we choose $m = 20$ KBytes, transmittable over the considered radio channels within short time limits (e.g., around 27 ms over a 6 Mbps DRSC channel).

The choice of the optimal number of hash functions $k$ improves the efficiency, at the cost of increased system complexity. In fact, to use a variable number of hash functions, the CA must transmit the used value of $k$ together with the filter. At the receiving side, the verifier must learn $k$ and use $k$ entries of a pre-established list of hash functions.

To overcome this complexity, we use a fixed number of hash functions $k$. Fig. 9(b) shows the case where $k = 10$ (for $m = 20$ KBytes), a compromise between computation complexity and false positives rate. We can see that the resulting false positives rate is reasonably low for small $n$ and converges to the performance provided by optimal values of $k$ when $n$ increases. As the
for the considerably high compression gain $c$ ($10 < c < 138$), it is independent of the number of hash functions $k$ (whether fixed or variable/optimal).

APPENDIX II

DETAILS OF LEAVE

A. Computation of the Eviction Quotient

An observing vehicle uses the following parameters to calculate the eviction quotient for vehicle $j$ (with accusations from different vehicles $i$) are the following:

- $A_i$ is the total number of accusations (issued by different vehicles) against vehicle $i$. $A_i$ is used to lower the weight of the accusations made by vehicle $i$ if it was already accused (i.e., it has lower credibility).
- $P_i$ is the accumulated sum of $|N_i|$, the number of $i$’s neighbors (as explained in Sec. VI-B) that the observing vehicle has encountered.
- $\alpha_i$ is the normalized value of the total number of accusations with respect to the total number of neighbors of vehicle $i$ ($0 \leq \alpha_i \leq 1$). This value is computed as follows: $\alpha_i = \frac{A_i}{P_i}$.
- $\omega_i$ is the weight of any accusation made by vehicle $i$. This weight depends on the number of accusations made against $i$: $\omega_i = 1 - \alpha_i$, giving $0 \leq \omega_i \leq 1$. Therefore, the weight of a node against which there are no accusations equals 1.
- $Q_j$ is the eviction quotient defining whether vehicle $j$ should be evicted. It is computed as follows: $Q_j = \frac{1}{P_j} \left( \sum_{i=1}^{P_j} \sigma_{ij} \omega_i \right)$, where $\sigma_{ij} = 1$ if there is an accusation against $j$ issued by $i$, and $\sigma_{ij} = 0$ otherwise.

The eviction quotient threshold ($Q_T$) is a configurable parameter. A typical value would be 0.5 (majority vote). If $Q_j > Q_T$, vehicle $j$’s messages are disregarded.

B. Resilience to Attackers

As LEAVE relies on nodes accusing attackers, there is a potential for abuse by attackers: A group of colluding attackers can accuse honest nodes and cause their eviction. In this section, we
analyze LEAVE’s resilience to such attacks. Assuming all nodes have roughly the same number of neighbors, \( P_j \approx P \), the eviction quotient becomes \( R_j = \frac{1}{P} \sum_{i=1}^{P} \sigma_{ij} \omega_i \). Let \( x \) be the fraction of attackers in the neighborhood. We can distinguish two cases:

**Case 1:** Attackers accuse all honest nodes; honest nodes do not accuse attackers. We consider this to be a strong attacker case. The eviction quotient required for a successful attack is:

\[
R = \frac{1}{P} \sum_{i=1}^{xP} 1 = x
\]  

(2)

**Case 2:** Attackers accuse all honest nodes; honest nodes accuse all attackers. We consider this to be a weak attacker case. The eviction quotient required for a successful attack is:

\[
R = \frac{1}{P} \sum_{i=1}^{xP} (1 - \frac{P - xP}{P}) = x^2
\]  

(3)

Fig. 10 shows the required percentage of attackers for a successful attack, given an eviction threshold of 0.5. We can see that, under the assumption of honest majority, LEAVE is highly resilient to colluding accusers.

**Figures**

![Diagram](image)

Fig. 1. Detection and Eviction Scheme Overview.
Fig. 2. Revocation of the Trusted Component (RTC) and Revocation using Compressed Certificate Revocation Lists (RC²RL).

Fig. 3. Misbehavior Detection System (MDS) operation.
Fig. 4. Performance of the MDS in the case of data anomalies.

(a) Detection of data anomalies

(b) Outlier detection by clustering

Fig. 5. The LEAVE protocol. Vehicle C has reached the eviction threshold for vehicle V, and broadcasts a disregard message. B is only in the transmission range of A, and gets the information from A when A reaches the threshold and sends a disregard message.
Fig. 6. Definition of the attacker’s neighborhood.
(a) Percentage of initially warned vehicles vs. vehicle density (in the FW scenario, the density is per lane)

(b) Average time to being warned vs. vehicle density

(c) Percentage of initially warned vehicles vs. average vehicle speed

(d) Average time to being warned vs. average vehicle speed

(e) Effect of the WRI parameter (AO scenario)

(f) Effect of the number of supporting signatures

Fig. 7. System performance vs. vehicle density, average speed, WRI, and number of supporting signatures.
Fig. 8. Bloom filter concept.

(a) Compression gain and false positives rate vs. filter size, (b) False positives rate vs. number of entries \( L \), using \( k=10 \) using optimal \( k \) hash functions

Fig. 9. Bloom filter performance.

Fig. 10. LEA VE: resilience to attackers.