SAMM: Situation Awareness with Machine Learning for Misbehavior Detection in VANET

Mohammed A. Abdelmaguid
School of Computing, Queen's University
Kingston, Ontario, Canada
18ma5@queensu.ca

Hossam S. Hassanein
School of Computing, Queen's University
Kingston, Ontario, Canada
hossam@cs.queensu.ca

Mohammad Zulkernine
School of Computing, Queen's University
Kingston, Ontario, Canada
mz@queensu.ca

ABSTRACT

Vehicular Ad hoc Network (VANET) is a foundation stone for connected vehicles. As vehicles' safety depends heavily on the exchanged data's accuracy, VANET has a low tolerance for false data. The process of intentionally exchanging inaccurate data is called misbehaving. Machine learning (ML)-based solutions were heavily invested in detecting misbehavior messages. However, they also have some limitations with respect to how much they can detect. To overcome such limitations, we introduce situation awareness (SA) as a powerful concept that can break the limits of the used ML models, leading to more accurate and reliable solutions. Situation awareness uses environmental elements and events to gain a holistic view of the system at any given time. In this paper, we propose using SA to predict the trust of the surrounding cars and consequently reevaluate the outcome of the used ML model. Based on the collected data and SA information, we may reject a message classified as benign by the ML model or vice versa. We used VeReMi dataset to evaluate the proposed approach called SAMM (Situation Awareness with Machine Learning for Misbehavior Detection in VANET) on different ML models with a wide range of features. The results show that the proposed approach improves the system's accuracy for various misbehavior attacks by enhancing the recall rate up to 24% and 50% in some cases.

CCS CONCEPTS

• **Security and privacy** → **Network security**.

KEYWORDS

VANET security, Situation Awareness (SA), Machine Learning (ML), VeReMi dataset, misbehavior detection, trust measurement.

ACM Reference Format:
Mohammed A. Abdelmaguid, Hossam S. Hassanein, and Mohammad Zulkernine. 2022. SAMM: Situation Awareness with Machine Learning for Misbehavior Detection in VANET. In *The 17th International Conference on Availability, Reliability and Security (ARES 2022), August 23–26, 2022, Vienna, Austria.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3538969.3543788

1 INTRODUCTION

Vehicle connectivity is becoming a monolith of road safety. The annual growth of connected vehicles reached 18.7% since 2016 [25]. Vehicles with Electronic Control Units (ECUs) communicate via various technologies to form Vehicular Ad Hoc Networks (VANETs). VANET is a self-organized network where vehicles connect in a decentralized manner, enhancing the driving experience and ameliorating safety and traffic management [6]. Different types of enabled communications have been utilized in VANET like Vehicle-to-Vehicle (V2V), Vehicles-to-Infrastructure (V2I), and generally, Vehicle-to-Everything (V2X) [4].

Consequently, security in connected vehicles has become a critical issue [4, 11]. VANET messages are broadcasted throughout the network, allowing everyone to share information with everyone. Hence, VANET nodes need to deal with distinguishing between malicious and benign messages [29]. Message authenticity and correctness are the main factors for achieving a secure and safe communication process. Message authentication has been a target for many researchers and has some standard protocols (IEEE 1609). Nevertheless, message correctness assurance remains a challenge.

Basic Safety Messages (BSMs) are one of the most critical exchanged messages in VANET. BSMs were defined by the Society of Automotive Engineers (SAE) in SAE J2735 standards [18]. Moreover, these messages are periodically shared and contain information exchanged between nearby vehicles. Specifically, they contain data such as speed, location, and time. Such information are used for various purposes like collision avoidance and traffic jam detection. Unfortunately, due to the public nature of BSMs, malicious nodes can broadcast false information, which can result in devastating consequences [12].

The open sharing environment in VANET led to dealing with a specific type of attack called misbehavior attacks, where vehicles share misleading information. Misbehavior messages are authentic messages that contain false information [33], and exchanging such erroneous information in VANET can propagate quickly. Additionally, cars and driver profiling may depend heavily on this information. Thus, assessing the trustworthiness of the surrounding nodes became integral because we need to guarantee the correctness of the received information and protect the driver's decision-making process from getting influenced by false information.

Content integrity requires critical and integral decision-making. However, traditional network security mechanisms tend to trust whoever manages to get inside the network. Hence, conventional all-or-none security mechanisms cannot deal with attacks aiming at message correctness. Moreover, trust evaluation has established itself as an asset for enabling secure and reliable exchanged information between network entities [17]. Accordingly, utilizing machine learning for attack classification and trust evaluation for inside attacks (e.g., misbehavior attack) can lead to a more secure environment in VANET.

Situation awareness (SA) has shown promising results in the field of Cyber Defense [20]. SA is about reaching the state of being aware of the circumstances around us. Primarily, we are interested in the events that are particularly relevant to the current situation. Ultimately, deploying situation awareness in VANET, especially for security, depends heavily on the environment and the behavior of the surrounding cars. Furthermore, SA relies massively on the amount and the quality of the provided data. Therefore, the more information you provide, the better your awareness of your surroundings.

This paper demonstrates using SA on ML models' results for detecting misbehavior attacks. In this context, SAMM quantifies situation awareness in VANET by calculating a trust score using collected information from the environment. Subsequently, the trust information is fed to the situation awareness model. The result is a misbehavior detection system that uses the powerful classification of ML and augments the accuracy of these results by utilizing the trust information obtained from the situation awareness model.

The main contributions of this paper are threefold:

- We introduce a novel approach for misbehavior detection based on situation awareness named SAMM.
- We evaluate our approach on a publicly available dataset and different types of machine learning models and demonstrated better results compared to the ones obtained with just the machine learning models.
- We contribute toward the standardization of VeReMi dataset by providing new baselines which future researchers can compare with.

The rest of the paper is organized as follows. In Section 2, the work related to misbehavior detection, machine learning, and situation awareness in VANET are discussed. In Section 3, we discuss the attack model along with the dataset and situation awareness. Section 4 provides a detailed description of SAMM, the misbehavior detection model using situation awareness. Section 5 highlights the simulation results of the proposed model. Finally, Section 6 concludes the paper and presents the future work.

2 RELATED WORK

2.1 Misbehaving Detection

Misbehavior detection focuses on identifying messages with incorrect or anomalous data. Misbehavior detection can be divided into node-centric or data-centric techniques [24]. A node-centric technique considers the data being correct if the sender can be trusted [31]. A Data-centric technique verifies the data itself to determine its correctness. We outline some of the existing work that discuss misbehavior detection. Bissmeyer et al. [3] proposed a way where message correctness verification depends on central decision-making authority. A central place receives the collected data and has the authority to make decisions regarding the nature of the data, whether normal or misbehaving. This approach suffers from dealing with a voluminous amount of data. The work by Grover et al. [7] can be considered as an example of a consensus method. In this scheme, a group of nodes tries to identify malicious nodes that perform any kind of misbehaving. Abu-Elkheir et al. [1] proposed a position verification scheme to help vehicles have more acumen for other vehicles' announced positions. They used direct and 2-hop neighbors to create a lower and upper boundary for the sender position, and if the sent position is beyond this plausibility area, a flag will be raised. In the absence of 2-hop neighbors, they used Received Signal Strength Indicator (RSSI) to verify an announced distance against relative distance to distance measures. The proposed scheme showed promising results for position verification in VANET.

Deploying a misbehavior detection system (MDS) for identifying internal attacks has shown promising results. Different works [9, 10] adopted ML-based MDS to detect falsification attacks. Khattab et al. [8] proposed an IDS which extracts features from a trace file to detect attacks in VANET. They used an artificial neural network (ANN) and fuzzified data to detect abnormal behavior of vehicles.

Van der Heijden et al. [31] shed light on the lack of generalized MDSs models due to the lack of a standard dataset for MDSs evaluation. Van et al. [32] published a labeled dataset called VeReMi for misbehavior attacks in VANET. VeReMi dataset aims to create a standardization platform upon which various misbehavior detectors can be evaluated. So et al. [28] produced ML-friendly features using VeReMi dataset. They compared the effectiveness of relying only on plausibility checks against using plausibility checks as amalgamated features within the ML models. Finally, they evaluated the models' ability to classify various attack types. In addition to adopting ML in MDS, trustworthiness measurement demonstrated how powerful a tool it could be for misbehavior detection [21].

2.2 Situation Awareness

Many different domains utilized situation awareness (e.g., robotics, psychology, AI, and computer science) [15]. However, utilizing SA for VANET security is still in its early stages. Hong et al. [14] proposed a situation awareness trust architecture (SAT). SAT addressed three aspects (trust attributes or policy control, proactive trust, and social trust). In the proposed architecture, trust attributes consist of entity trust and data trust. Also, they showed how to set up the trust in advance by using the vehicular network instead of waiting for a particular event to occur. Finally, they proposed using social networks in the absence of Road Side Unite (RSU) infrastructure by using the surrounding vehicles' assessment of the situation. Thus, this work proposes utilizing situation awareness for establishing trust in VANET, but it lacks real or simulated experiments, unlike our proposed work.

The work in [19] investigated the network security situation as a nonlinear time series. It used a radial basis function (RBF) neural network as it can achieve a nonlinear combination between the input space and the output space. In this case, the input is previously-stored values about the history and current situation, while the output is the following state situation with different time windows. The time windows represent the selected number of continuous situation values (e.g., setting 3, 4, or 5 continuous situation values). The simulation experiment showed how the RBF neural network achieved better results than the backpropagation (BP) neural network predicting the network security situation. However, it may be improved significantly using an optimization algorithm and adjusting the RBF neural network structure.

Hashem et al. [13] explored the use of SA in VANET from the perspective of improving traffic safety. By utilizing the exchanged information through communication channels like basic safety messages in VANET, they can be aware of the state of the network around the hour. In particular, they utilized SA to maintain the used routing links' stability and help the quality of service (QoS). The routing algorithm updates the calculated QoS metrics if one established route becomes ineffective.

3 PRELIMINARIES AND PROBLEM FORMULATION

3.1 Dataset and Attack Model

VeReMi dataset is publicly available and actively maintained on GitHub, ensuring its integrity. There are five different attacks represented in VeReMi: constant position, constant offset, random position, random offset, and eventual stop. VeReMi contains 225 simulations with three different traffic densities (low, medium, and high). Every vehicle that receives a message creates a log file that consists of the sender vehicle ID, a unique message ID, position (x,y,x), RSSI, and velocity (x,y,z). One of the reasons for using VeReMi dataset is its recent attention in VANET security study. Mitchel et al. [22] discussed the need for a standardized dataset so that future researchers can validate their work. So et al. [27, 28] used a modified version of VeReMi to create a ML-friendly model of the metrics that we used in one of our experiments.

3.2 Plausibility Checks and ML for Detecting Misbehavior

One of the main approaches used recently for message correctness is the adoption of plausibility checks. Mainly, plausibility checks are used as a feature vector to enhance the ML models' classification capabilities [28]. In our experiments, we used a number of features and plausibility checks. Most of them were introduced in the previous works [27, 28]. The used plausibility checks are as follows: (1) displacement in X and Y positions, (2) difference in X and Y velocities, (3) acceleration in X and Y, (4) predicted position in X and Y, (5) predicted velocity in X and Y, (6) difference between calculated and predicted position in X and Y, (7) difference between calculated and predicted velocity in X and Y, (8) distance between X and Y, and (9) distance between predicted X and Y.

3.3 Situation Awareness

Situation awareness (SA), in its basic form, is about knowing what's happening in your vicinity. Figure 1 shows both the three phases of SA and its adoption in the context of VANET security for detecting misbehavior attacks. In the first phase (perception), we rely on the data represented in BSMs. Then, in the second step (comprehension), we use the collected data to calculate plausibility checks, feed them to the ML model, and provide it to the trust equation. Finally, in the last step (projection), we consider the ML and the trust model scores to assess the situation at that particular moment and predict the imminent situation's development.

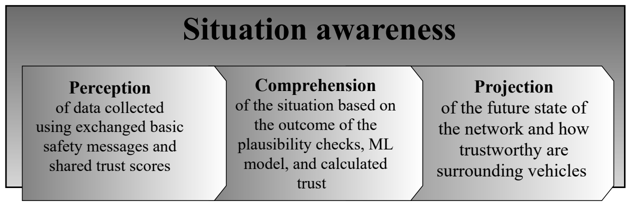
Situation awareness
Perception of data collected using exchanged basic safety messages and shared trust scores
Comprehension of the situation based on the outcome of the plausibility checks, ML model, and calculated trust
Projection of the future state of the network and how trustworthy are surrounding vehicles

Figure 1: Situation awareness phases in VANET trust model

One major challenge in SA is how to measure it. Quantitative analysis is used to represent the network security situation state at one point in time [16]. While quantitative analysis can provide warnings about the network security situation, it suffers from the lack of information regarding the true nature of the attack. Nevertheless, some well-known techniques for measuring situation awareness are inherited from military aviation testing, like subjective ratings, simulation freeze, and task performance measures. In our case, we are extending one of these measuring techniques-subjective ranking technique (SART)- to apply it in the security domain. In the subjective ranking, ranking a driver is based on the ranking by observers or other drivers. In our work, we similarly measure situation awareness by giving a score for the vehicle in the form of a trust score built upon other drivers' reported observations.

3.4 Trust in VANET

Ensuring the integrity of exchanged data in VANET is a must. It is more critical for BSM data. Trust management systems in VANET use related information as values for trust functions or as inputs for the trust inference models. The trust score or the inference results can determine the trust level of a certain vehicle. Accordingly, our model used the collected data from basic safety messages along with data from the environment as values for the trust calculation function, Logistic Trust (LT) [2].

Logistic Trust (LT) model was adopted for measuring SA in terms of how trustworthy the vehicles in the network are [2]. LT is very compatible with SA as it is time and experience-driven. SA tends to collect data iteratively over time, which contributes heavily to the LT trust scoring process. Logistic trust model consists of the following three parameters:

Dissatisfaction: It measures the contradiction of the received information from vehicle s. IF $i_n(s)$ is the number of incorrect messages s sent and $v_n(s)$ is the total number of sent messages by s, then dissatisfaction in time-slot n is

$$\delta_n(R,s) = \frac{i_n(s)}{v_n(s)} \tag{1}$$

while the total dissatisfaction up to time slot n is

$$\bar{\delta}_n(R,s) = \frac{\sum_{n=1}^{n} \delta_n(R,s)}{n} \tag{2}$$

Flag: In the original LT, the flag parameter is part of the direct trust. Also, if the dissatisfaction score was higher than a certain pre-declared threshold β, a flag $\alpha_n(R, s)$ is raised for this s vehicle. We modified this part by making the flag part of the indirect trust and raising the flag after a certain number of attacks. We did this because LT was designed for Ad hoc networks in general; this made the flag parameter more tolerant of misbehaving. Originally, the flag used to be lowered if the dissatisfaction score went below the threshold after more interactions. In VANET, the cars do not have that long interaction time to interact to the point where the dissatisfaction score can get higher and lower over a long period. Therefore, we found that cars should be flagged after a certain number of attacks. In the event that a flag was raised by mistake, the overall trust score does not solely depend on the flag state. Since other parameters (e.g., expectation, dissatisfaction) may result in accepting the received messages from that car.

Expectation: $\rho_n(R, s)$ is the mean of the recommendations regarding vehicle s in time slot n

$$\rho_n^{LT}(R, s) = \frac{b_n(s) + 1}{b_n(s) + g_n(s) + 2} \tag{3}$$

where bad recommendations are represented as $b_n(s)$, and good recommendations are represented as $g_n(s)$. Altogether, the final trust score is given by

$$t_n(R, s) = \frac{1}{1 + e^{-}(\vec{V}.\vec{W} + W_0)} \tag{4}$$

where

$$\vec{V} = \{\bar{\delta}_n(R, s), \rho_n^{LT}(R, s), \alpha_n(R, s), t_{n-1}(R, s)\}$$

In our version of LT, the $\vec{V}$ is a vector of the values of the four main components of the trust equation: dissatisfaction, flag, expectations, and the trust score for the vehicle s in the previous time slot.

From the above discussion, we can formulate the problem into two parts. First, we need to measure SA in terms of a trust score based on LT. Second, we have to integrate ML with SA. The first part will help utilize SA with ML and find a way to incorporate both in the decision-making process. The second part will lead to overcoming ML limitations and improving its detection rate.

4 PROPOSED APPROACH

In this section, we describe the proposed situation awareness and machine learning approach for misbehavior detection. In the proposed approach, we have two main goals: measuring situation awareness to reflect the state of the network in a given time using logistic trust, and adopting this holistic view with the acquired information from the used machine learning model to achieve better results in detecting misbehavior attacks.

4.1 Measuring SA Using LT

In order to measure SA, we utilized the Subjective Ranking Technique (SART) concept [26]. In SART, observers in the network can rate a driver. Following the same concept in VANET, other drivers can act as observers to rank a certain vehicle. Logistic Trust (LT) [2] was used for ranking vehicles by calculating their trust score.

In SAMM, when vehicle R receives a message from vehicle s, LT calculates the trust score for vehicle s using two sources of

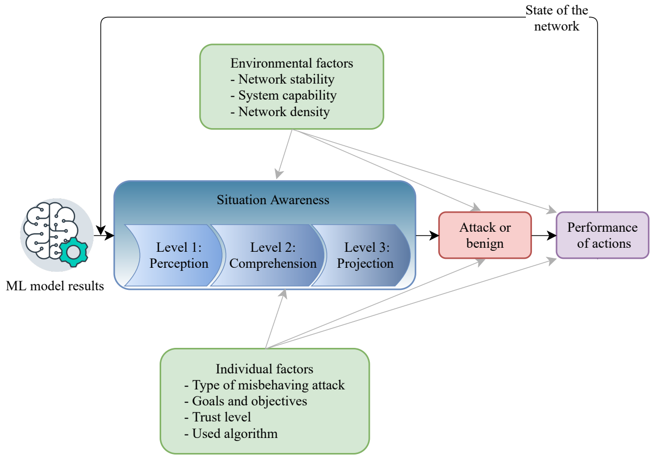

Figure 2: Situation awareness and machine learning for detecting misbehavior attacks in VANETs

information: Vehicle R's direct experience with vehicle s and indirect experience between other vehicles and vehicle s. LT uses three main parameters, namely, dissatisfaction, flag, expectation, for trust calculations. In Section 3, we explained how each parameter is calculated. In this section, we will present how we adopted these parameters in our approach.

4.1.1 Dissatisfaction. $\delta_n(R, s)$ is a measure of the inconsistency of the received information from a certain vehicle. We measure dissatisfaction by calculating the difference between the correct and incorrect received messages from vehicle s. Initially, we create a counter for correct and incorrect messages for every vehicle. Then, every time we mark a message as benign, we increase the correct message counter by one and vice versa. After that, we use Equation 1 to calculate the total dissatisfaction in the current time slot.

4.1.2 Flag. $\alpha_n(R, s)$ is related to the number of attacks detected from a certain vehicle s. A flag is raised if a vehicle exceeds a certain threshold in terms of the number of attacks.

4.1.3 Expectation. $\rho_n(R, s)$ is a reflection of what other vehicles in the network think about vehicle s. We look at reported attacks from other vehicles as bad recommendations $b_n(s)$ and non attack interactions as good recommendations $g_n(s)$. Consequently, we calculate expectation according to Equation 3

4.2 Integrating SA with ML for Misbehaving Detection

In this section, we discuss the main components of the SA model shown in Figure 2.

4.2.1 System input: Generally, in SA systems, the input is the state of the environment. SA was integrated with the purpose of enhancing the detection rate of ML models, and the input here is the output of ML models. Another difference is the way we build the used ML models. For the purpose of building the ML model in a way that is related to the environment, plausibility checks are mandatory for the ML building process. Plausibility checks use the

Algorithm 1 SAMM algorithm

Inputs:
$Message_n$: Current received message
Car_n: Sender vehicle
$\overrightarrow{A}$: Weight vector for the trust vector variables
$attackProb$: The ML model probability that a message is an attack message
$nonAttackProb$: The ML model probability that a message is a non attack message
$dissatisfactionValue$: The dissatisfaction value
$dissatisfactionThreshold$: The dissatisfaction threshold
$attackThreshold$: The threshold of the message being an attack
$notAttackThreshold$: The threshold of the message being a non attack
$trustAttackThreshold$: The threshold of the trusting score for not accepting the message
$trustnotAttackThreshold$: The threshold of the trusting score for accepting the message
$flagThreshold$: The threshold for raising $flag$

Outputs:
$Misbehaving_n$: Message n is a misbehaving message
$Benign_n$: Message n is a benign message

Variables:
$trustScore_n$: Vehicle current calculated trust score
$flag_n$: A flag which indicates whether the sender is an attacker or not

```
if $Message_i$ is detected as an attack and $Car_n$ is not in $attackerList$ then
    add $Car_n$ in $attackerList$ and attack counter is initialized to 1

/*Calculate the trust score using the modified logistic trust model*/
if $Message_n$ is detected as an attack message by the ML model then
    if $Car_n$ number of attacks > $flagthreshold$ then
        set $flag_n$ to 1

    $attackMessagesPercentage$ = ($receivedAttackesMessages$ / $receivedMessages$) * 100
    if $attackMessagesPercentage > dissatisfactionThreshold$ then
        set $dissatisfactionValue$ to 1
    else
        set $dissatisfactionValue$ to 0

    $expectationValue$ = ($receivedAttackesMessages$ + 1) / ($receivedAttackesMessages$ + $receivednonAttackesMessages$ + 2)
    $trustScore_n = 1/1 + e^{\overrightarrow{V}.\overrightarrow{W}+W_0}$

    /*Modifying the model output based on the trust score*/
    if $trustScore_n < trustAttackThreshold$ and $nonAttackProb > notAttackThreshold$ then
        $Benign_n$
else
    if $Message_i$ is detected as a non attack and $Car_n$ is not in $nonAttackerList$ then
        add $Car_n$ in $nonAttackerList$ and a non attack counter is initialized to 1
    else
        increase the non attack counter by 1
    if $trustScore_n < trustnotAttackThreshold$ and $attackProb > attackThreshold$ then
        $Misbehaving_n$
```

collected information from the environment, which is crucial for SA. We also did not take the output of the ML in terms of 0 or 1 but in terms of probabilities. Thus, it gives us a better understanding of how the ML model assisted the situation by providing a percentage of the message being benign or malicious. Another input here is "Feedback", and its role is to enhance the results in the future. We build the current decision in time n, considering the previously collected and computed information in the time slot $n - 1$. Also, in the LT, we include the previous trust score in calculating the trust score in time n.

4.2.2 SA Level 1 - Perception: To better detect misbehavior attacks, perception starts with collecting the surrounding vehicles' data stored in the BSMs. Such data include the vehicle's location, speed, and send time. Another source of information is the outcome of machine learning. The outcome is the probability of whether the received message is an attack. The general rule in SA is that more information leads to better predictions.

4.2.3 SA Level 2 - Comprehension: Misbehavior detection needs to analyze the provided data to improve upon perception. For instance, we calculate the trust elements of the trust function as the ratio of false and benign messages or other vehicles' recommendations about a certain vehicle to compute its reputation.

4.2.4 SA Level 3 - Projection: The projection step concerning misbehavior detection leads to the capability of estimating other vehicles' level of trust. Accordingly, this is achieved through the build-up knowledge of vehicles' behavior and the awareness of the current situation (outcomes of SA level 1 and level 2). In level 3, we determine the most trustworthy cars and vice versa. Also, we identify the vehicles that showed signs of misbehaving before and see if they exceeded the defined threshold of misbehaving. Ultimately, this will be used in the next step to enable the most suitable decision in the current time to be taken. Hence, it determines whether to accept the received message or not.

4.2.5 Environmental factors: Various elements like network stability and density affect the situation awareness process. Additionally, the deployment of sensors and security software also plays a role in the environment where the system operates. System factors assist on different levels of SA and cooperate in the decision-making process. In our case, we assume that network stability and density are fixed, and they are the same for all vehicles.

4.2.6 Individual factors: There exist individual or group attributes that influence the SA process [23]. Individual goals and objectives may change according to the environment, which leads to new plans to meet these new goals. In our case, the system operator may change the trust acceptance threshold for a better detection rate. In addition, the operator can apply a different model for measuring situation awareness other than the LT model. Moreover, the operator can decide to target different types of misbehavior attacks.

4.3 SA with ML Algorithm for Detecting Misbehavior Attacks

As shown in Algorithm 1, after receiving a classification result regarding a recently received message from a surrounding vehicle (line 4), we either confirm this classification or change it based on SA decision. The algorithm makes use of the data collected from the environment by calculating dissatisfaction (lines 9 and 11), expectation (line 12) and raising flags (line 6) according to the recent situation. Finally, we combine two main sources of information: the ML model decision and the calculated information based on the environment, to reach a final decision. ML model information comes in the form of probability, not discrete decisions. Therefore, instead of identifying whether this is an attack message, we are more interested in the probability of this message being an attack or benign. Furthermore, the environment information is used to reflect SA and predict the surrounding cars' trust value. Accordingly, we use these two values (classification probabilities and trust value) to either keep the initial classification obtained from the ML model regarding the received message or change it (line 16 and line 23). Finally, we end up either accepting a message which was classified as an attack or vice versa (lines 16 and 23).

The algorithm's complexity is $O(n)$. Even though we search multiple lists (e.g., attacker list, non-attacker list), we still do not have any nested loops. Accordingly, integrating SA should not pose a considerable overhead.

5 EXPERIMENTS AND RESULTS

We conducted four different experiments to understand the improved aspects and their nature after utilizing SA. The first experiment (Section 5.1) is different from the other three and can be considered a pre-experiment. The pre-experiment gives an idea about the effectiveness of the SA algorithm itself in general.

We conducted the other three experiments on ML models to demonstrate the influence of SA on the ML detection rate for misbehavior attacks. We used a publicly available dataset named VeReMi for these experiments to test a wide range of misbehaving attacks. We established the ground truth ourselves in the second experiment (Section 5.2). Therefore, we built the ML model and calculated the plausibility checks in order to make SA the only variable in the equation. Thus, we compared the detection results between the ML model and the ML model with SA to investigate the impact SA has on the accuracy rate of detecting misbehavior attacks. The third experiment (Section 5.3) helped us refine our understanding of the effect of SA on identifying misbehaving. Furthermore, we compared this work against a very recent work [30] that used federated learning on top of ML instead of SA, as in our case. Finally, in the fourth experiment (Section 5.4), we further demonstrated the influence of SA on enhancing ML models' capabilities in detecting misbehavior attacks.

5.1 Investigating the Correlation Between SA Calculated Trust and Attack Density

This experiment was conducted to observe the correlation between the measured trust expectations for the vehicles in the network and their actual behavior over time. Therefore, we have the calculated trust value and the reported attacks based on ML.

We used a scatterplot to represent the correlation between the inverted trust value and the number of detected attacks for each vehicle. Figure 3 shows the correlation in all five attacks in addition to the correlation coefficients at the top left corner. Each dot in the plot represents a vehicle. A dot's location depends on the number of detected attacks for this vehicle and the calculated trust score. The error region is also illustrated in Figure 3 as the area in light blue. This error region corresponds to the 95% confidence interval of the estimate.

Positively correlated variables move in the same direction as shown in the graph. We used the inverse of the trust value to simplify the scatterplot's visualization between calculated trust and detected attacks. Also, the correlation coefficient on the top of the graph reflects the association between two variables. Variables with a correlation coefficient magnitude over 0.5 are considered moderately correlated. In this experiment, the minimum obtained correlation was 0.59, and we reached 0.71 in the random offset attack. The scatterplots for all the attacks show a small gap between the trust scores. The gap is not considerable, but it resulted from the outlier nodes (i.e., cars with a small number of attacks with high trust scores and cars with a large number of attacks with low trust scores). We added an area around the nodes (the small blue circle) related to this node's number of detected attacks and a color bar to show where most of the nodes lay and how small the outliers' numbers are. The gap may look smaller in the first three attacks, but the scale is too small. Accordingly, "smal" here does not reflect more outliers covering the area because the difference is insignificant. Primarily, this shows how the calculated trust results correlate to the cars' behavior in the actual scenario. However, this experiment was an early stage one which we considered as a proof of concept for our assumption. Thus, we cannot interpret the result coherently as to how the correlation coefficients and the types of attacks are related. However, this will be done in the subsequent experiments.

5.2 Analyzing the Effect of Using SA on ML Attack Detection Rate

To fathom the merging between ML and SA, we needed to study the outcome of the plain ML model after synthesizing SA. For this experiment, we designed the ML model and the plausibility checks. Consequently, we made sure to make SA as the only variable in the equation.

5.2.1 Parsing dataset. We started with importing all VeReMi `.json` files. Each file belongs to one vehicle and contains the received BSMs from its neighbors in a 300-meter range. Also, attack files are categorized into three different density scenarios: low (35 - 39 vehicles), medium (97 - 108 vehicles), and high (491 - 519 vehicles).

Because SA reflects the system situation at a specific time, we made sure to sort all the records in chronological order. Most importantly, this is different than most of the used approaches out there, where they mainly focus on pairing the sender and the receiver messages. This pairing process results in changing the records' chronological order. Of course, the traditional approach is not wrong, but it only focuses on creating attack features without considering time as one of the elements. After that, we converted all the files to `.csv` file format without concatenating them together.

5.2.2 Plausibility checks and supporting features. We calculated eight different plausibility checks and features which we represent in Section 3. Some features were calculated as plausibility checks,

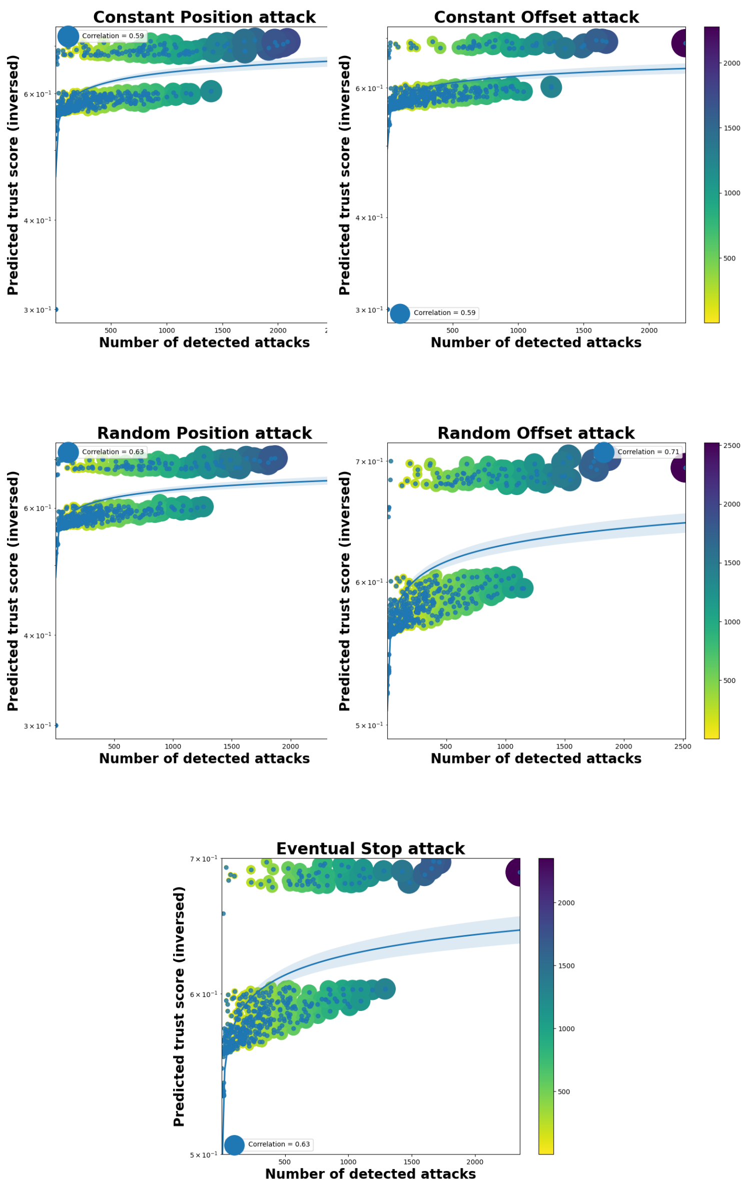

Figure 3: Correlations between SA trust and attack density

while others were calculated to gain more insight into the environment. Subsequently, feature numbers 1, 6, 8 are the plausibility checks, while the others act as supporting features. We needed the supporting features to expand the understanding of the surrounding cars' behavior and use them in calculating plausibility checks (e.g., calculating acceleration a to calculate position later).

5.2.3 Random forest and trust calculation. Many researchers adopted machine learning models for anomaly detection [5]. Besides, the random forest classifier has shown better results in terms of high accuracy, especially while dealing with extensive data.

The first and second experiments were performed on Python 3. We used the scikit-learn library (`sklearn.ensemble`) for the Random forest classifier. We used 0.75 of the data for training and the rest for testing. The number of the used trees in the Random forest classifier is 100. After training the model, we assessed and saved the accuracy score for each one of the five attacks. Afterward, Algorithm 1 identifies the effect of including SA on the final results.

Table 1 demonstrates the difference in performance between the obtained results from the ML models and the ones after including SA in all three different traffic density scenarios. We observe a significant increase in the total accuracy rate on the various types of attacks in different densities, except for the first attack in the high density. It was possible due to the simplicity of this attack. Using displacement as a plausibility check for this attack makes detection accurate, almost near 100%, and any small change would significantly affect the result. However, SA considers outside elements, not just the plausibility checks, influencing the outcomes. Another reason behind such behavior is the type of injected information. To make our approach as generic as possible, we chose not to filter the provided information based on the targeted attack. Therefore, we prevented the data from being fixed to only what we tried to detect. Still, more than 90% of the cases show a very promising increase in the detection rate. Finally, we can see how integrating SA has enhanced the overall detection rate except for the attacks that do not require further inference, such as constant position. Nevertheless, some attacks such as constant positions are straightforward attacks that can be detected using the traditional machine learning approaches. Therefore, while SA does not contribute more in detecting such simple attacks, it complements the conventional approach to detecting more complex attacks.

5.3 Examining the Effect of Using SA on Recall and Precision in ML

In this experiment, we investigated the enhancement in attack detection between the recently published approach in [30] and ours. Researches adopting ML for misbehavior detection systems focus on the ML models or feature detection. In this work, we are trying to push the limits of ML for attack detection in VANETs by introducing the idea of integrating situation awareness. Similarly, the work in [30] enhanced the straightforward ML models in VANETs for privacy-preserving against misbehavior detection. They proposed using Federated Learning (FL) for misbehavior detection without sharing BSM data with a third party, achieving better precision, recall, and performance. Their work studied the effect of using FL versus traditional ML models, while ours studies the effect of SA over traditional models. Consequently, the experiments followed their approach of comparing the basic ML model against the “add to” version of it.

We constructed two sub-experiments. We operated on the same dataset (VeReMi) and implemented their neural network. In the first sub-experiment, we chose the plausibility checks to use for the model. Still, the only difference between the two tested works was the used features and plausibility checks, as we consider them as part of the SA model. In the second sub-experiment, we left everything the same, even the used features. The idea behind this experiment is to observe the effect of SA just on the model itself.

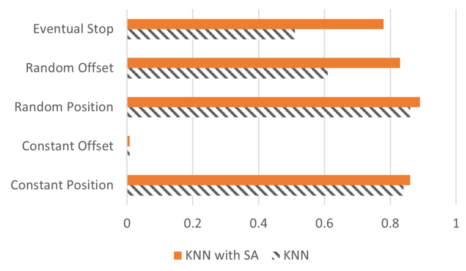

Figure 4: Recall comparison between the obtained results by the KNN model and after integrating SA

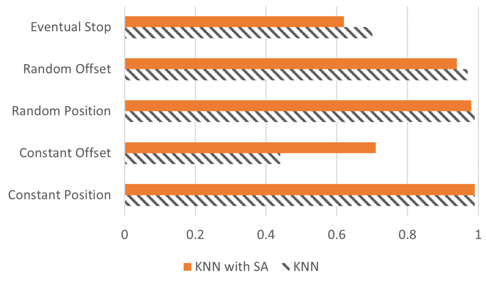

Figure 5: Precision comparison between the obtained results by the KNN model and after integrating SA

In terms of evaluation, we followed their precision and recall evaluation matrix as shown in Table 2. This will give us more insights into the behavior of SA on ML results. Given that precision and recall affect each other, we found out that SA enhances recall even if it is at the cost of precision. For Attacks 1, 3, and 4 in Table 2a we observe a notable increase in recall in return for the small sacrifice in precision. Even for the second sub-experiment, we notice a boost in recall in all the different attacks, as shown in Table 2b at the expense of a minor reduction in precision, as mentioned.

5.4 Improvement in the Recall by Utilizing SA

Finally, we evaluate our approach by comparing our results with the results obtained by So et al. [28], where they show how they managed to increase the detection of misbehavior using ML and plausibility checks. Unlike the previous experiments, the used dataset is a special version of VeReMi called ML-friendly VeReMi [28]. In this version, all the files of one scenario were concatenated together in one `.csv` file. Moreover, two attributes were added to each `.csv` file named Attacker Type and Receiver Type. “Attacker Type” attribute contains a number from 0 to 5, where 0 indicates that this is a normal vehicle and the rest of the numbers reflect one of the five attacks in the dataset. The mapping between the number and the attack type was obtained from the ground truth file, represented in the original dataset. Moreover, “Receiver ID” was added to build a sender-receiver pair that helps represent each sender’s journey.

We did not have to implement the ML model for this experiment as the code was available [28]. Instead, we added SA to the process

Traffic density	Low		Medium		High	
	ML	ML & SA	ML	ML & SA	ML	ML & SA
Constant position	0.982%	0.983%	0.97%	0.98%	0.999%	0.612%
Constant offset	0.859%	0.878%	0.807%	0.837%	0.83%	0.86%
Random position	0.849%	0.861%	0.88%	0.89%	0.91%	0.92%
Random offset	0.8436%	0.861%	0.813%	0.842%	0.87%	0.89%
Eventual stop	0.867%	0.906%	0.904%	0.922%	0.91%	0.92%

Table 1: Comparison between ML detection results with/without SA in different traffic densities

Precision					
	Constant pos.	Constant offset	Random pos.	Random offset	Eventual stop
FL Model	0%	-1%	0%	1%	-1%
FL & SA	-2%	9%	1%	-4%	9%
Recall					
FL Model	-13%	-4%	0%	0%	-16%
FL & SA	4%	0%	8%	15%	0%

(a) The enhancement in terms of precision and recall by the Federated learning model and our model (sub-experiment #1)

Recall					
	Constant pos.	Constant offset	Random pos.	Random offset	Eventual stop
FL Model	-13%	-4%	0%	0%	-16%
FL & SA	3%	11%	8%	7%	10%

(b) The enhancement in terms of recall by the Federated learning model and our model where same plausibility checks and features were used (sub-experiment #2)

Table 2: Examining the effect of using Situation Awareness (SA) on recall and precision in ML

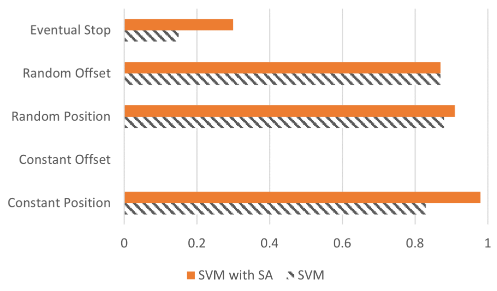

Figure 6: Recall comparison between the obtained results by the SVM model and after integrating SA

to see the effect on the ML model's final results. However, Figures 4, 5, 6, and 7 show the obtained results (for the KNN, SVM, KNN with SA, and SVM with SA). We did not find a significant difference between their published results and the ones we got. We had to compare with the new results to measure the impact of SA on them. The paper represented two different experiments; one was done

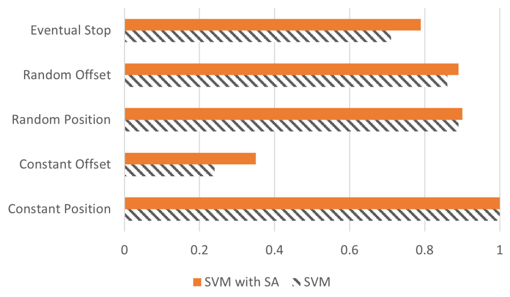

Figure 7: Precision comparison between the obtained results from the SVM model and after integrating SA

using the KNN model and the other using SVM. For the KNN model, Figures 4 and 5 show the increase in recall, especially for attacks 4 and 5, where they achieved an increase in the recall by 36% and 52% with only 3% and 11% decrease in precision. Furthermore, for the SVM model, Figures 6 and 7 show how we achieved an increase in both recall and precision in all attacks.

In Figures 4 and 6, we find a small recall rate in the constant offset attack both for the KNN and the SVM models. Nevertheless, this is a detection problem in the first place. Looking at how the same plausibility checks have been used for both models, we can see how we ended up with the same behavior regarding detecting the same attack (constant offset attack). Accordingly, better plausibility checks for detecting this attack are needed in general. Still, under those circumstances, we can see that SA maintained the same behavior of improving the precision rate when the recall is saturated as shown in Figures 5 and 7. Ultimately, SA can only improve recall when there is room for improvement in the first place.

6 CONCLUSION

Misbehavior attacks are one of the biggest challenges to achieving a secure and reliable Vehicular Ad hoc Network (VANET). With all of their advantages, ML models require a new perspective to perform better and acclimate to VANET characteristics. One of those characteristics is the ability to share personal information using Basic Safety Messages (BSMs) in an open environment. In this work, we used a publicly available dataset VeReMi. We showed how utilizing SA with traditional ML model approaches can lead to better detection of misbehavior attacks. Furthermore, Logistic Trust (LT) was used to measure SA. By incorporating SA with ML, we ascertained that the ML model's classification results still have room for improvement (by accepting a message labeled as an attack and vice versa) to obtain a better detection rate. Moreover, we found that SA increases the network sensitivity to detect misbehaving. Accordingly, we noticed how this manifested in a better recall rate, which eventually increased the detection rate. The performance of the introduced approach was evaluated through extensive experiments. The proposed approach showed promising results with over 50% increase in recall in some cases. It has been evident that utilizing SA with ML for misbehavior detection in VANET is promising for achieving a better detection rate and keeping the network secure and reliable.

REFERENCES

[1] Mervat Abu-Elkheir, Sherin Abdel Hamid, Hossam S. Hassanein, Ibrahim M. Elhenawy, and Samir Elmougy. 2011. Position verification for vehicular networks via analyzing two-hop neighbors information. In *2011 IEEE 36th Conference on Local Computer Networks*. 805–812.

[2] Saneeha Ahmed and Kemal Tepe. 2016. Misbehaviour detection in vehicular networks using logistic trust. In *2016 IEEE Wireless Communications and Networking Conference*. 1–6.

[3] Norbert Bißmeyer, Joël Njeukam, Jonathan Petit, and Kpatcha M. Bayarou. 2012. 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*. 73–82.

[4] Tamás Bécsi, Szilárd Aradi, and Peter Gáspár. 2015. Security issues and vulnerabilities in connected car systems. In *2015 International Conference on Models and Technologies for Intelligent Transportation Systems*. 477–482.

[5] Nebrase Elmrabit, Feixiang Zhou, Fengyin Li, and Huiyu Zhou. 2020. Evaluation of machine learning algorithms for anomaly detection. In *2020 International Conference on Cyber Security and Protection of Digital Services*. 1–8.

[6] Mario Gerla, Eun-Kyu Lee, Giovanni Pau, and Uichin Lee. 2014. Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds. In *2014 IEEE World Forum on Internet of Things (WF-IoT)*. 241–246.

[7] Jyoti Grover, Nitesh Kumar Prajapati, Vijay Laxmi, and Manoj Singh Gaur. 2011. Machine Learning Approach for Multiple Misbehavior Detection in VANET. In *Advances in Computing and Communications*. 644–653.

[8] Anna Gruebler, Klaus D. McDonald-Maier, and Khattab M. Ali Alheeti. 2015. An intrusion detection system against black hole attacks on the communication network of self-driving cars. In *2015 Sixth International Conference on Emerging Security Technologies (EST)*. 86–91.

[9] Sohan Gyawali and Yi Qian. 2019. Misbehavior detection using machine learning in vehicular communication networks. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. 1–6.

[10] Sohan Gyawali, Yi Qian, and Rose Qingyang Hu. 2020. Machine learning and reputation based misbehavior detection in vehicular communication networks. *IEEE Transactions on Vehicular Technology* (2020), 8871–8885.

[11] Talal Halabi, Omar Abdel Wahab, Ranwa Al Mallah, and Mohammad Zulkernine. 2021. Protecting the Internet of Vehicles Against Advanced Persistent Threats: A Bayesian Stackelberg Game. *IEEE Transactions on Reliability* (2021), 970–985.

[12] Talal Halabi and Mohammad Zulkernine. 2019. Reliability-driven task assignment in vehicular crowdsourcing: A matching game. In *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. 78–85.

[13] Mahmoud Hashem Eiza, Thomas Owens, Qiang Ni, and Qi Shi. 2015. Situation-Aware QoS Routing Algorithm for Vehicular Ad Hoc Networks. *IEEE Transactions on Vehicular Technology* (2015), 5520–5535.

[14] Xiaoyan Hong, Dijiang Huang, Mario Gerla, and Zhen Cao. 2008. SAT: Situation-Aware Trust Architecture for Vehicular Networks. In *Proceedings of the 3rd International Workshop on Mobility in the Evolving Internet Architecture*. 31–36.

[15] Martin Husák, Jana Komárková, Elias Bou-Harb, and Pavel Čeleda. 2018. Survey of attack projection, prediction, and forecasting in cyber security. *IEEE Communications Surveys & Tutorials* (2018), 640–660.

[16] Martin Husák, Jana Komárková, Elias Bou-Harb, and Pavel Čeleda. 2018. Survey of attack projection, prediction, and forecasting in cyber security. *IEEE Communications Surveys & Tutorials* (2018), 640–660.

[17] Rasheed Hussain, Jooyoung Lee, and Sherali Zeadally. 2020. Trust in VANET: A survey of current solutions and future research opportunities. *IEEE Transactions on Intelligent Transportation Systems* (2020), 2553–2571.

[18] IEEE. 2006. IEEE Trial-Use Standard for Wireless Access in Vehicular Environments - Security Services for Applications and Management Messages. *IEEE Std 1609.2-2006* (2006), 1–105.

[19] Yang Jiang, Cheng-hai Li, Li-shan Yu, and Bo Bao. 2017. On network security situation prediction based on RBF neural network. In *2017 36th Chinese Control Conference (CCC)*. 4060–4063.

[20] Yu-Beng Leau and Selvakumar Manickam. 2015. Network security situation prediction: a review and discussion. In *International Conference on Soft Computing, Intelligence Systems, and Information Technology*. 424–435.

[21] Qin Li, Amizah Malip, Keith M. Martin, Siaw-Lynn Ng, and Jie Zhang. 2012. A reputation-based announcement scheme for VANETs. *IEEE Transactions on Vehicular Technology* (2012), 4095–4108.

[22] Robert Mitchell and Ing-Ray Chen. 2014. A survey of intrusion detection techniques for cyber-physical systems. *ACM Computing Surveys (CSUR)* (2014), 1–29.

[23] Cyril Onwubiko. 2016. Understanding Cyber Situation Awareness. *International Journal on Cyber Situational Awareness* (2016), 11–30.

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

[25] Juniper Research. 2020. *In-Vehicle Commerce Opportunities Drive Total Connected Cars to Exceed 775 Million by 2023.* Retrieved June 2, 2020 from https://www.juniperresearch.com/press/press-releases/in-vehicle-commerce-opportunities-exceed-775mn

[26] Stephen J Selcon and RM Taylor. 1990. Evaluation of the Situational Awareness Rating Technique(SART) as a tool for aircrew systems design. *AGARD, Situational Awareness in Aerospace Operations* (1990), 23–53.

[27] Steven So, Jonathan Petit, and David Starobinski. 2019. Physical Layer Plausibility Checks for Misbehavior Detection in V2X Networks. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*. 84–93.

[28] Steven So, Prinkle Sharma, and Jonathan Petit. 2018. Integrating Plausibility Checks and Machine Learning for Misbehavior Detection in VANET. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. 564–571.

[29] Joey Sun, Shahrear Iqbal, Najmeh Seifollahpour Arabi, and Mohammad Zulkernine. 2020. A classification of attacks to In-Vehicle Components (IVCs). *Vehicular Communications* (2020), 100253.

[30] Aashma Uprety, Danda B. Rawat, and Jiang Li. 2021. Privacy Preserving Misbehavior Detection in IoV Using Federated Machine Learning. In *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*. 1–6.

[31] Rens Wouter van der Heijden, Stefan Dietzel, Tim Leinmüller, and Frank Kargl. 2018. Survey on misbehavior detection in cooperative intelligent transportation systems. *IEEE Communications Surveys & Tutorials* (2018), 779–811.

[32] Rens W. van der Heijden, Thomas Lukaseder, and Frank Kargl. 2018. VeReMi: A Dataset for Comparable Evaluation of Misbehavior Detection in VANETs. In *Security and Privacy in Communication Networks*. 318–337.

[33] Sherali Zeadally, Ray Hunt, Yuh-Shyan Chen, Angela Irwin, and Aamir Hassan. 2012. Vehicular Ad-hoc Networks (VANETS): status, results, and challenges. *Telecommunication Systems* (2012), 217–241.