J. Asiat. Soc. Bangladesh, Sci. **51**(1): 123-150, June 2025
DOI: http://doi.org/10:3329/jasbs.v51i1.82796

A TRUST-BASED MALICIOUS RSU DETECTION MECHANISM IN EDGE-ENABLED VEHICULAR AD HOC NETWORKS

FARHANA SIDDIQUA AND MOSARRAT JAHAN*
Department of Computer Science and Engineering, University of Dhaka, Dhaka-1000, Bangladesh

Abstract

Edge-enabled Vehicular Ad Hoc Networks (VANETs) provision real-time services, storage, computation, and communication facilities to vehicles through Roadside Units (RSUs). Nevertheless, RSUs are often easy targets for security assaults due to their resource-constrained nature and placement in an open, unprotected environment. The compromised RSUs impede the VANET operations, causing traffic mismanagement and threats to human safety. Hence, an effective malicious RSU detection mechanism is crucial for VANETs. More specifically, a mechanism to detect the misbehavior of RSUs on RSU-to-RSU (R2R) communications, essential for message forwarding, beacon message sharing, and traffic alert sharing among RSUs, needs to be included. Besides, current works use only vehicle speed and density in beacon messages to assess trust without considering the sensor-detected data in the same messages. Nonetheless, sensor data is useful for traffic management, and neglecting them creates inaccuracy in trust estimation. This paper addresses these limitations and proposes a trust-based scheme to detect malicious RSUs that uses R2R interaction to analyze an RSU's behavior. We also offer a mechanism to detect alteration of sensor-detected data in beacon content and incorporate this scheme in the trust calculation of RSUs. The experimental results show that the proposed solution effectively detects approximately 92% malicious RSUs, even in the presence of hostile vehicles. Moreover, integrating the proposed solution with the VANET routing protocols improves routing efficiency.

Key words: Vehicular Ad Hoc Networks, VANET, Roadside Unit (RSU), Trust Management, Security, Beacon Message

I. Introduction

Vehicular Ad Hoc Network (VANET) is a leading-edge technology enabling systematic management of vehicles running on roads and highways. It models the transportation system as an ad hoc network and facilitates information exchange among the moving vehicles (Onieva *et al.*, 2019). It improves human safety, reduces road accidents and traffic jams, and creates provisions for smart travel planning (Sheikh *et al.*, 2019). Due to the highly dynamic environment of VANET, the availability of correct information at the right moment is a prime requirement for its proper operation. In this regard, Roadside

*Corresponding author: mosarratjahan@cse.du.ac.bd

Units (RSUs) play a vital role in accelerating information processing and providing services at low latency. However, the advantage of minimum latency comes with the cost of numerous security and privacy issues introduced by RSUs, affecting information accuracy (Abhishek *et al.*, 2019).

RSUs are edge devices usually deployed along the roadside on traffic lights, bus stops, road signs, etc., to provide various services to the vehicles (Onieva *et al.*, 2019) They can be installed anywhere but are cost-effective to deploy in areas where traffic volume is high, and placement of expensive general-purpose edge devices with higher computing facilities is not feasible (Onieva *et al*., 2019). In VANETs, RSUs are trusted with several important responsibilities such as vehicle authentication (Yao *et al.*, 2019), rogue vehicle detection (Al-Otaibi *et al.*, 2019), and revocation (Malik *et al.*, 2018). However, RSUs fall short of serving their purposes competently for two reasons. Firstly, they are typically resource-constrained compared to the general-purpose edge devices, although they have enough resources to serve the vehicles in their coverage area (Onieva *et al.*, 2019). Due to their resource-constrained nature, they cannot support computation-intensive security mechanisms, making them easy victims of various security attacks. Secondly, due to the outdoor placement without tight protections from network operators, RSUs are vulnerable to intrusions, physical attacks, malfunctions, node compromise, sensor tampering attacks, etc. (Van der Heijden *et al.*, 2018). Therefore, RSUs compromised by security attacks severely affect the correct functioning of the entire system, leading to severe consequences threatening human safety. Hence, accurate identification of malicious RSUs and avoiding them from the VANET operation is essential to ensure safe, secure and time-sensitive operation of VANETs.

Traditional cryptography-based security mechanisms are not suitable for dynamic-nature VANET due to their time-consuming and intensive computations and the inability to address some security attacks such as false data injection and internal attacks (Zaidi *et al*., 2014). Hence, trust-based schemes have been considered as an alternative to identify malicious entities in VANET at a low-cost (Hussain *et al.*, 2020). At present, very few research works focus on identifying the malicious behavior of RSUs, an indispensable part of VANETs (Abhishek *et al.*, 2019; Lu *et al.*, 2018; Alnasser and Sun, 2021). Among them, (Abhishek *et al.*, 2019) detects malicious RSUs based on vehicles' feedback (vehicle-to-RSU (V2R) communication), (Lu *et al.*, 2018) based on Received Signal Strength Indicator (RSSI), and (Alnasser and Sun, 2021) based on the discrepancy of an RSU's decision with other RSUs. However, none of them consider the behavior of an RSU on R2R communication. Besides providing services to vehicles and other road entities, RSUs also interact with their one-hop neighbor RSUs for message routing

(Mershad *et al.*, 2012), periodic beacon message broadcasting (Maglaras *et al.*, 2013), and traffic alert sharing (Jindal and Bedi, 2017). They use message routing to forward vehicles' data packets (Mershad *et al.*, 2012) and utilize beacon messages to share traffic information with other RSUs and vehicles (Maglaras *et al.*, 2013). Besides, RSUs share traffic alerts to avoid unwanted situations such as accidents and bad road conditions (Jindal and Bedi, 2017). This indicates that R2R transmissions occupy a significant portion of the communications that an RSU uses to contact the VANET entities. Hence, the trustworthiness of an RSU should also reflect its reliable behavior in all these aspects, which is missing in the existing literature. In contrast to V2R communication, R2R communications are stable as the positions of RSUs are static, and they frequently interact with their one-hop neighbor RSUs, enabling precise and error-free trust calculations.

Besides, existing works verify vehicle speed and density to determine the legitimate beacon content, and these parameters are used in the trust calculation based on beacon messages (Arshad *et al.*, 2018; Zaidi *et al.*, 2015). Apart from vehicle speed and density, an RSU also shares sensor-detected data such as humidity (Jindal and Bedi, 2017), temperature (Jindal and Bedi, 2017), and carbon emission level (Maglaras *et al.*, 2013) in beacon messages. The correctness of these sensor-detected data is also crucial as they directly impact traffic management; for example, vehicles usually try to avoid industrial areas prone to excessive carbon emissions. Hence, ignoring sensor-detected data to verify the validity of beacon content can create difficulty in traffic management. Therefore, trust calculation based on the beacon content should also reflect the correctness of sensor data, which is not considered in the literature.

In this paper, we address the shortcomings mentioned above and propose a malicious RSU detection mechanism based on trust calculations that evaluates an RSU's behavior depending on its interaction with other RSUs in the VANET. In particular, we make the following contributions:

- We propose a trust-based mechanism to assess an RSU for its behavior in all the R2R communications. We incorporate an equation to compute an aggregated trust score of an RSU that uniquely combines its score in individual R2R communication.
- We offer a robust mechanism to detect the correctness of sensor-identified data in a beacon message and assign a weight to each beacon message based on the validity of its sensor data. Finally, sensor data verification is combined with the verification of vehicle speed and density in the same beacon message to compute trust based on beacon content.
- We implement the proposed scheme and evaluate the performance through extensive

experiments. The results demonstrate the effectiveness of the proposed solution in detecting malicious RSUs, which is approximately 92% in the presence of rogue vehicles. The experimental results also show that the proposed scheme improves the routing efficiency of the existing VANET routing protocols when incorporated with them. Besides, we observe that sensor data verification in beacon messages moderately improves the decision accuracy of the proposed scheme.

II. Related Work

Trust-based mechanisms are cost-effective solutions to detect malicious entities in VANETs. They assign trust scores to VANET entities based on their behavior to measure their credibility (Hussain *et al.*, 2020; Soleymani *et al.*, 2015). Although there are numerous works on the trust mechanisms for detecting malicious vehicles (Hussain *et al.*, 2020; Soleymani *et al.*, 2015; Tripathi and Sharma, 2019), very few papers consider the issue of identifying rogue RSUs.

(Abhishek *et al.*, 2019) proposed a trust-based mechanism where every vehicle sends feedback about the RSU it has interacted with to a central trusted server. In this regard, a vehicle evaluates an RSU based on the channel quality and the total number of packets received from or transmitted to the RSU. The central trusted server calculates an aggregated trust value for every RSU based on the received feedback that is later compared with a threshold value to classify the RSU as malicious or authentic. They also employed a Gaussian kernel-based similarity metric mechanism to handle the impact of false feedback from misbehaving vehicles. This work only considered downlink packet drops; therefore, the authors proposed an updated version in (Abhishek and Lim, 2022) that defends uplink attacks and downlink attacks. However, this model only handles selective packet modification attacks performed by RSUs during V2R communication, which partly reflects the behavior of an RSU. Besides, (Lu *et al.*, 2018) used the physical (PHY)-layer properties such as RSSI of the ambient radio signal to detect rogue edge nodes. In this scheme, resource-limited smartwatches and smartphones inside a car outsource heavy computation to an edge node located inside the vehicle. In this case, an outside malicious edge node situated in the VANET environment can launch a man-in-the-middle attack by sending messages to the mobile devices requesting services. The mobile device uses the physical layer properties to distinguish ambient radio signal traces of an outside edge node from an inside legitimate edge node. However, this solution cannot ensure the content accuracy shared by the edge devices, which is crucial for the reliable operation of the VANET. Moreover, (Hao *et al.*, 2008) proposed a distributed key

management scheme where a trusted authority plays the role of the key generator and an RSU as a key distributor. An RSU forms a group with the vehicles within its transmission range and provides them the group key after confirming vehicles' authenticity. After receiving any complaints about other misbehaving vehicles, the trusted authority takes help from the RSU to recover the malicious vehicle's real identity. In this case, a compromised RSU might provide the signature of a legitimate vehicle instead of the malicious one to the authority. The proposed scheme prevents this issue by not providing RSU any access to vehicles' private keys. The main goal of this work is to ensure that an RSU performs its duty accurately as a key distributor not to identify the rogue RSUs. (Alnasser and Sun, 2021) proposed a trust model to prevent malicious RSUs that conduct recommendation attacks while detecting rogue road entities such as vehicles, cycles, motorcycles, and pedestrians. In this model, every VANET entity observes their one-hop neighbors' behavior and sends their observations to the nearest RSUs. The responsible RSU analyzes the received behavioral information and decides on an entity's trustworthiness. Besides, it sends the final list to the central cloud server, which monitors RSUs' behavior and makes the final decisions regarding malicious road entity detection. In contrast to the previous works, our proposed solution analyzes the behavior of an RSU in all the means it can communicate with other RSUs in the VANET and produces an aggregated trust value reflecting realistic conjecture on the reliability of an RSU for R2R communications.

Existing research works that explore false data detection mechanisms in VANETs mainly consider vehicles' speed and density shared in beacon messages. For example, (Arshad *et al.*, 2018) proposed a trust management system and fake data detection scheme that utilizes vehicle speed and density shared in beacon messages to measure the trustworthiness of a vehicle. This scheme also uses beacon and safety messages to filter out incorrect messages and gets facts from data to assess traffic data reliability. Besides, (Zaidi *et al.*, 2015) proposed an Intrusion Detection System (IDS) that also uses speed and density collected from neighbor vehicles' beacon messages to identify rogue vehicles. The proposed IDS analyzes the collected data statistically to detect false information attacks. In another work, (Al-Otaibi *et al.*, 2019) classified traffic data using vehicle speed to identify rogue vehicles. In this work, an RSU analyzes traffic data provided by vehicles within its transmission area to calculate an estimated speed range. A vehicle is rogue if its speed does not belong to the calculated speed range. Similarly, (Paranjothi *et al.*, 2020) also utilized vehicles' speed to detect rogue vehicles. This scheme chooses vehicles with a more significant number of neighboring nodes as guard vehicles. These vehicles perform a hypothesis test on the neighbor vehicles' speed to

identify malicious vehicles. Moreover, (Liu *et al.*, 2020) proposed a false message detection scheme that uses a traffic flow model to analyze the actual behavior of a vehicular environment by utilizing vehicle speed and density collected from neighbor vehicles' beacon messages. (Jalooli *et al.*, 2024) utilized Blockchain to store trust scores of vehicles. This scheme validates sensor data to determine message authenticity. It further allows vehicles to query RSUs to get information on the trustworthiness of message sender vehicles. Besides, (Zhang *et al.*, 2024) used both vehicle consensus and Gradient Boosting Decision Tree (GBDT) to detect false messages where vehicle densities are processed as time series data. (Lone and Verma, 2025) proposed a mechanism to detect misbehavior in VANETs based on multidimensional plausibility and consistency checks on beacon data, focusing on position and speed information.

In contrast to prior works, our scheme considers sensor-detected data as well as speed and density information in the beacon messages to verify the validity of traffic data and incorporates both types of data to evaluate trust based on beacon content.

III. Proposed Scheme

In this section, we present a trust-based solution to detect malicious RSUs in the VANET. We first discuss the system model for the proposed scheme, followed by a detailed description of the working principle of the proposed scheme.

A. System Model

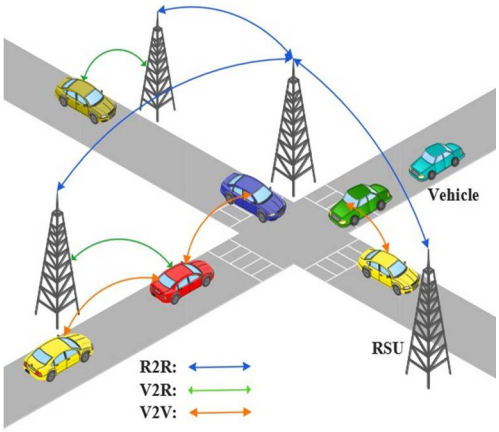


Fig. 1. System model of the proposed scheme.

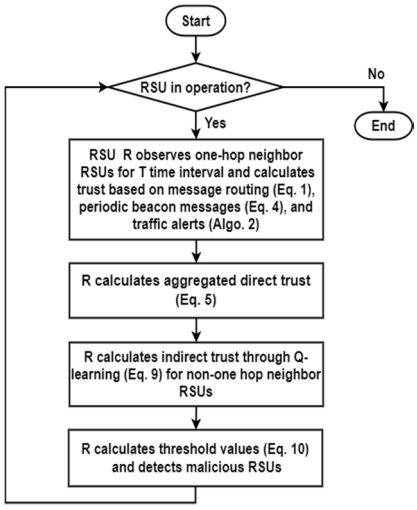


Fig. 2. Flow diagram of the proposed scheme for RSU *R*.

Figure 1 presents the system model of our proposed scheme. It comprises two entities, *vehicles* and *RSUs*. *Vehicles* are equipped with an Onboard Unit (OBU), a Global Positioning System (GPS), and different types of sensors to collect information from their surroundings (Onieva *et al.*, 2019). They exchange periodic beacon messages, which is also known as Basic Safety Message (BSM) (Van der Heijden *et al.*, 2018) and Cooperative Awareness Message (CAM) (Jin and Papadimitratos, 2018) to inform their existence and provide traffic information perceived through their sensors. They also share traffic alerts to notify emergency events to other vehicles and the nearest RSU. The communication among the vehicles known as Vehicle-to-Vehicle (V2V) communication ranges from 50 to 300 meters (Zhang and Chen, 2019). On the other hand, the communication between an RSU and vehicles is known as V2R communication. *RSUs* are local cloud servers placed at less than the one-kilometer distance in an area where traffic volume is usually high (Zhang and Chen, 2019). They are equipped with a network device supporting IEEE 802.11p protocol, devices to communicate with the infrastructure network, a GPS, and sensors (Van der Heijden *et al.*, 2018). They provide real-time services to vehicles and process the data collected from vehicles through V2R communications (Al-Otaibi *et al.*, 2019). Besides, RSUs generate beacon messages periodically (Maglaras *et al.*, 2013) and share alert messages when emergency events occur (Jindal and Bedi, 2017). They also work as a relay node to propagate messages generated by vehicles and RSUs (Mershad *et al.*, 2012). An RSU communicates with other RSUs via R2R communication, and the communication range is limited to 1000 meters (Zhang and Chen, 2019). We consider a hop count of 6 to transmit messages generated by RSUs as traffic data can be relevant up to $5km$ (Lee *et al.*, 2014) and the maximum distance between two RSUs is $1km$.

B. ***Overview of the Proposed Scheme***

Each RSU R monitors its one-hop neighbor RSUs for a pre-defined time duration T shown in Fig. 2. During this time, R observes the behavior of its one-hop neighbor RSUs for routing messages, broadcasting beacon messages, and transferring traffic alerts. When T expires, R assigns trust scores to its neighbors in each of the above-mentioned communication scenarios based on their behavior and eventually combines all the trust scores to calculate the direct trust of the one-hop neighbor RSUs. For non-one hop RSUs, R uses the Q-learning mechanism (Guleng *et al*., 2019) to determine their trust values. R also decides a threshold value for every other RSUs in the network and compares it with the corresponding trust value to identify an RSU as *legitimate* or *compromised* node.

*C. **Trust Calculation based on Message Routing***

The proposed scheme observes the behavior of an RSU for dropping packets and altering packet content while it acts as a relay node to route messages from a source vehicle to a destination vehicle (Mershad *et al.*, 2012), or forward packets from remote RSUs to a central RSU (Huang *et al.* 2017). Each RSU R_1 forwards packets to its one-hop neighbor RSU R_2 and monitors the behavior of R_2 on forwarding these packets for a fixed time interval *T*. R_1 counts the number of forwarded and dropped packets by R_2 and checks whether the forwarded messages are maliciously modified. Based on the observation, R_1 calculates the trust value, $Trust_{routing}$ of R_2 as follows:

$$Trust_{routing} = \frac{P_{forward}}{P_{forward} + P_{drop}} \times P_{modify} \quad (1)$$

where $P_{forward}$ is the number of packets forwarded, P_{drop} is the number of packets dropped, and P_{modify} is the packet modification parameter. Eq. 1 indicates that $Trust_{routing}$ is computed based on packet forwarding ratio ($P_{forward}/P_{forward} + P_{drop}$) and packet modification parameter (P_{modify}). Here, P_{modify} is a binary variable where $P_{modify} = 0$ when an RSU is maliciously altering packet content and $P_{modify} = 1$ for an honest RSU not changing packet content. The multiplication operation in Eq. 1 puts higher priority on P_{modify} as $P_{modify} = 0$ makes $Trust_{routing} = 0$. This argument is justified as the trust value of a malicious RSU modifying packet content should be 0 even though it is forwarding some data packets. $Trust_{routing}$ varies in the range [0, 1].

Each RSU R_1 uses a Watchdog module (Marti *et al.*, 2000) that overhears the incoming and outgoing traffic of other entities within R_1's transmission range. Hence, a watchdog module of R_1 can detect whether R_2 forwards a packet towards the next node. This module stores the recently sent packets by R_1 and removes a packet from the buffer when it overhears the same packet being forwarded by the next-hop RSU R_2. In this case, R_1 increments $P_{forward}$. If a packet remains in the buffer for more than the expected time $t_{expected(R_1, R_2)}$, R_1 considers R_2 has dropped that packet and increments P_{drop}. R_1 calculates $t_{expected(R_1, R_2)}$ as follows (Bhoi, and Khilar, 2014):

$$t_{expected(R_1, R_2)} = \frac{L}{r_{(R_1, R_2)}} + \frac{d_{(R_1, R_2)}}{V_{propagation}} + t_{other} \quad (2)$$

where *L* is the length of message, $r_{(R_1, R_2)}$ is the data transmission rate, $d_{(R_1, R_2)}$ is the distance between R_1 and R_2, $V_{propagation}$ is the propagation speed, and t_{other} represents the queuing and processing delay. The watchdog module in R_1 estimates t_{other} from the packet forwarding tendency of R_2 that can be computed by taking the

difference of packet reception time and packet forwarding time in R_2. Besides, R_1 adjusts the link data rate based on the value in the *window* field of the TCP header in acknowledgement packets.

The watchdog mechanism also compares the hash value of a packet on the incoming interface of the observed RSU with the hash value of the same packet on the outgoing interface (Patil and Tahiliani, 2014). The hash values are computed on the packet fields that are not supposed to change during routing (Patil and Tahiliani, 2014). If both hash values are the same, then no packet modification is performed by next-hop RSU. If the watchdog module detects a packet modification then R_1 sets P_{modify} = 0, otherwise P_{modify} = 1.

D. ***Trust Calculation based on Beacon Messages***

Each RSU periodically transmits *hello* messages that are beacon messages to its one-hop neighbor RSUs and vehicles to inform its existence and traffic-related information (Maglaras *et al.*, 2013). The proposed scheme considers the beacon message generation rate and the accuracy of beacon message content to detect malicious behavior of RSUs. An RSU is trustworthy only when it does not cause flooding attacks and propagate false information using beacon messages. Hence, the proposed scheme first examines whether any flooding attack occurs. If a flood attack is not detected, trust value for beacon messages is computed by analyzing the correctness of beacon content.

1) ***Trust Calculation based on Beacon Message Generation Rate:*** A malicious RSU prevents the message propagation by other entities of the VANET by flooding the communication channel with beacon messages. We use the flooding attack detection mechanism proposed in (Sajjad *et al.*, 2015) to detect RSU's misbehavior. An RSU R_1 observes its one-hop RSU R_2 in a time slot i of length T and counts $B_i(R_2)$, the number of beacon messages generated by R_2 during this interval. R_1 also keeps track of the beacon message rate of R_2 for the latest Z time slots and calculates the weighted average of beacon message generation rate as $B_{avg}(R_2) = \sum_{t=1}^{Z}(t/Z) \times B_t(R_2)$. If $B_i(R_2) > B_{avg}(R_2)$ in a time slot *i*, then flooding attack is detected, and R_1 sets $Trust_{beacon}$ to 0; Otherwise $Trust_{beacon}=1$.

Sender ID	Position	Timestamp	Event Type	Event Value	Location

Fig. 3. Vehicle and RSU traffic alert format.

2) ***Verification of Beacon's Content****:* Each RSU periodically shares beacon messages with its adjacent RSUs and vehicles within its transmission range to notify its existence, traffic condition, weather forecast, etc. (Jindal and Bedi, 2017). Besides, vehicles also

share beacon messages with their neighbor vehicles and the nearest RSU. Alongside, an RSU collects data about traffic situations, weather, etc. through its sensors (Sheikh *et al.*, 2019). Thus, an RSU receives huge volume of data from the beacon messages provided by both vehicles and neighbor RSUs and from its sensors (Hussain *et al.*, 2020). The RSU analyzes those data, generates aggregated results for different purposes, and includes the analyzed result in the beacon messages to provide a traffic overview to the vehicles and one-hop neighbor RSUs. An RSU usually shares speed (Al-Otaibi *et al.*,

Algorithm 1. Content verification of the *i*-th beacon message.

Input:
$TH_{speed\_density}$: threshold to verify speed and density
TH_{time}: threshold to verify timestamp
n: number of beacon messages received in T
Speed: estimated average speed by R_1
Density: estimated vehicle density by R_1
$B[i].Speed$: speed in R_2's i-th beacon
$B[i].Density$: vehicle density in R_2's i-th beacon
M_V: traffic alert received from a vehicle
$X[n]$: list of vehicle count reporting *IGNORE_RSU*
$Beacon[n]$: list of beacon message status

```
1:  function VERIFY_BEACON_CONTENT(R2, B[i])
2:      Y = N = 0
3:      if |B[i].Speed − Speed| ≥ TH_speed_density then
4:      Beacon[i] = 0
5:      else if |B[i].Density − Density| ≥
        TH_speed_density then
6:      Beacon[i] = 0
7:      else
8:      Beacon[i] = 1
9:      end if
10:     if Beacon[i] = = 1 then
11:     while M_V ! = NULL do
12:     if M_V.Event_Type = =' IGNORE_RSU'
13:     and M_V.Event_Value = = 0 and
14:     B[i].Timestamp − M_V.Timestamp ≤ TH_time
15:     and M_V.Location == R2.Position then
16:         Y + +
17:     else
18:         N + +
19:     end if
20:     end while
21:     end if
22:     if Y > N then
23:         Beacon[i] = 0
24:         X[i] = Y
25:     else if Y = = 0 then
26:     Total ← count_adjacent_vehicle()
27:     X[i] = Total
28:     else
29:         X[i] = Y
30:     end if
31: end function
```

2019), density (Arshad *et al.*, 2018), temperature (Jindal and Bedi, 2017), humidity (Jindal and Bedi, 2017), and carbon emission level (Maglaras *et al.*, 2013) in beacon messages. If flooding attack is not detected (discussed in Section III-D1), R_1 verifies the content of the i-th beacon message received from R_2 in two ways as described in Algo. 1. They are:

- R_1 estimates the speed and density of vehicles coming from the area of R_2 (Al-Otaibi *et al.*, 2019; Arshad *et al.*, 2018) and matches them with the same information in R_2's beacon message. If they vary by a certain threshold value, $TH_{speed\_density}$, R_1 sets the i-th beacon message, $Beacon_i = 1$; Otherwise, R_1 sets $Beacon_i = 0$ shown in lines 3∼9 of Algo. 1.
- In line 10, if $Beacon_i = 1$ (i.e, the speed and density information are correct) R_1 counts the feedback of intermediate vehicles to verify the remaining data in R_2's i-th beacon message. A vehicle V in the transmission range of R_2 verifies temperature, humidity, and carbon emission level shared in R_2's beacon message through its sensors. If the discrepancy of information in R_2's beacon message and V's sensor data exceeds a threshold value, TH_{sensor_data}, V generates an *IGNORE_RSU* traffic alert indicating invalid content. The neighbor vehicles of V verify the generated alert as they have also received the same beacon message from R_2. They share V's alert with their neighbor vehicles if the alert is correct. Otherwise, they discard the traffic alert. Thus, the alert propagates through the VANET and ultimately reaches the one- hop neighbor RSU R_1. R_1 receives alerts from multiple adjacent vehicles and counts those IGNORE_RSU alerts M_V whose timestamp difference with R_2's i-th beacon is less than or equal to a pre-defined threshold value TH_{time} in lines 10∼21. If the majority of neighbor vehicles ($Y > N$ in line 22) agree that R_2's beacon content is inaccurate, R_1 sets $Beacon_i = 0$, otherwise $Beacon_i = 1$. Here, $X[i]$ keeps a record of the number of adjacent vehicles of R_1 reporting IGNORE_RSU for the i-th beacon and used to calculate a weight w_i in Eq. 3. If R_1 does not receive any alert, $X[i]$ is set to the number of adjacent vehicles in lines 26∼27 in Algo. 1. Note that in Algo. 1, R_1 verifies speed, density, and sensor data one by one and considers neighbor RSU R_2 legal ($Beacon_i = 1$) if all of these parameters are accurate. However, R_2's beacon content is considered invalid if R_1 finds modification in any of these parameters and does not proceed to check another one.

In our scheme, both vehicles and RSUs generate traffic alerts following the format shown

in Fig. 3 (Arshad *et al.*, 2018). Here, *Sender ID* is a unique ID of a vehicle or RSU, *Position* is the current position of a vehicle or RSU, *Timestamp* is the traffic alert creation time, *Event Type* indicates traffic event, *Event Value* contains a binary value to indicate the presence and absence of the event and *Location* denotes the event place. We define a new event type *IGNORE_RSU* to detect invalid data. A vehicle assigns *Event Value* = 0 for *IGNORE_RSU* to indicate an RSU at *Location* is a malicious RSU generating false data and 1 to confirm it as an honest RSU.

3) ***Trust Calculation based on Beacon Content:*** Suppose R_1 receives *n* beacon messages from R_2 during *T*. R_1 receives *IGNORE_RSU* alerts from adjacent vehicles and determines $Beacon_i$ based on majority vehicles' opinion or the result of its verification of speed and density as shown in Algo. 1. R_1 assigns a weight w_i to each beacon message *i* as follows:

$$w_i = \frac{X_i}{\sum_{i=1}^{n} X_i} \tag{3}$$

where X_i is the total number of one-hop neighbor vehicles of R_1 reporting *IGNORE_RSU* alert for the *i*-th beacon message. w_i actually indicates the proportionality of adjacent vehicles of R_1 reporting IGNORE_RSU alerts for the *i*-th beacon to the total number of adjacent vehicles generating IGNORE_RSU alerts for *n* beacon messages in *T*. If $Beacon_i$ = 1 (i.e, the speed and density information are correct) and no alert is received for the *i*-th beacon message in sensor data verification phase (line 25 of Algo. 1), X_i is set to the total number of vehicles adjacent to R_1 to prevent w_i setting to 0 and assign highest possible trust score to an RSU not altering beacon content. The trust value based on beacon content is calculated as the weighted average of $Beacon_i$ generated during *T* as $\sum_{i=1}^{n}(w_i \times Beacon_i)$. Here, w_i is multiplied with $Beacon_i$ to assign a score on R_1's belief on the correctness of the *i*-th beacon message. Finally, the trust based on beacon message is computed as follows:

$$Trust_{beacon} = \begin{cases} 1/0, for\ flooding\ attack \\ \sum_{i=1}^{n}(w_i \times Beacon_i), \quad otherwise \end{cases} \tag{4}$$

E. ***Trust Calculation based on Traffic Alerts***

Traffic alerts are time-sensitive, and erroneous traffic alerts can cause serious problems such as difficulties in managing traffic and creates threats to human safety (Zaidi *et al.*, 2014). When an emergency event occurs, vehicles observing that event generate and broadcast traffic alerts to the vehicles within their transmission range and the nearest RSU (Arshad *et al.*, 2018). Similarly, RSUs noticing the same event also generate and broadcast the traffic alerts to the one-hop neighbor RSUs and vehicles under their coverage (Ahmed *et al.*, 2018). A malicious RSU can modify the traffic alert with

malicious intent. Thus, the trust of an RSU should also reflect the degree of RSU's capability of transmitting authentic traffic signals.

Figure 4 presents a scenario of how traffic alert is verified in the proposed scheme. Here, RSU R_2 monitors the area where an accident took place and RSU R_1 is a one-hop neighbor of R_2 observing the behavior of R_2 for T time duration. Vehicles observing the accident notify R_2 about the event and broadcast the traffic alert to vehicles within their transmission range. Besides, R_2 also broadcasts a traffic alert for the same accident to R_1

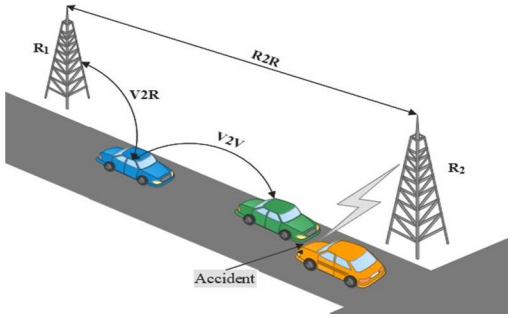


Fig. 4. R2R false alert detection.

Algorithm 2. Trust calculation of R_2 based on traffic alerts.

Input:
M_{R_2}: traffic alert received from one-hop RSU R_2
$M_V[n]$: traffic alert received from n adjacent vehicles
TH_{time}: threshold value for timestamp differences

1: $i = 0$
2: $Y = N = 0$
3: **while** $i < n$ **do**
4: **if** $M_{R_2}.Event\_Type == M_V[i].Event\_Type$ and $M_{R_2}.Event\_Value == M_V[i].Event\_Value$ and $M_{R_2}.Location == M_V[i].Location$ and $M_{R_2}.Timestamp - M_V[i].Timestamp \leq TH_{time}$ **then**
5: $Y++$
6: **else**
7: $N++$
8: **end if**
9: $i++$
10: **end while**
11: **if** $Y > N$ **then**
12: $Trust_{alert}=1$
13: **else**
14: $Trust_{alert}=0$
15: **end if**

and vehicles under its transmission range. Thus, a vehicle receives traffic alerts on the same event from neighbor vehicles and RSUs. It verifies the authenticity of the received traffic alerts using the information sensed by itself or the same traffic alerts received from other neighbor vehicles (Guleng *et al*., 2019) and RSUs. A nonsource vehicle considers the content of the maximum number of received alerts on the same event as valid and shares that message with its neighbors. Thus, the alert message propagates from one vehicle to another and ultimately reaches R_1. RSU R_1 receives traffic alerts on an event from R_2 and multiple adjacent vehicles V. It calculates the trust value of R_2 following Algorithm 2. For each received traffic alert $M_V[i]$ from an adjacent vehicle, R_1 compares it with the traffic alert M_{R_2} received from R_2. If both messages match on (1) Event Type, (2) Event Value, (3) Location (as shown in Fig. 3) and the difference of timestamps is less than or equal to a pre-defined threshold TH_{time}, R_1 considers them as a match and increments Y. Otherwise, N is updated. If the number of match, Y is greater than the number of nonmatch, N, then R_1 sets the trust of R_2 based on traffic alert, $Trust_{alert}$ to 1; Otherwise $Trust_{alert} = 0$, indicating R_2 as a malicious RSU. We can accomplish the same traffic alert verification process using the Decentralized Environmental Notification Message (DENM) (Santa *et al*., 2014) in Intelligent Transportation System (ITS).

*F. **Aggregated Direct Trust Calculation***

R_1 calculates the direct trust, $Trust_{direct}$ of R_2 as follows:

$$Trust_{direct} = (w_1 \times Trust_{routing} + w_2 \times Trust_{beacon}) \times Trust_{alert} \quad (5)$$

where $Trust_{routing}$, $Trust_{beacon}$, and $Trust_{alert}$ are trust values based on message routing, beacon message broadcasting, and traffic alert sharing, respectively. We multiply $Trust_{alert}$ with the weighted sum of $Trust_{routing}$ and $Trust_{beacon}$ to assign highest priority to traffic alerts. Among VANET applications, traffic alert-based services are of utmost importance, serving time-critical and emergency functions such as the notification of an accident, and bad road conditions (Zaidi *et al*., 2014). Improper timing of traffic alerts can cause severe consequences such as road accidents hampering human safety. VANET entities use periodic beacon messages to enhance traffic efficiency and collaboration (Al-Otaibi *et al*., 2019). Meanwhile, message routing mechanisms are used to provide other services like social networking and infotainment (Mershad *et al*., 2012). Here to note that $Trust_{alert}$ holds a binary value (1 or 0). Due to non-critical timing nature, we assign

weights to $Trust_{beacon}$ and $Trust_{routing}$ based on the frequency of respective messages. w_1 and w_2 are weights that sum to 1 and are defined as follows:

$$w_1 = \frac{F_{routing}}{F_{routing} + F_{beacon}} \quad (6) \qquad w_2 = \frac{F_{beacon}}{F_{routing} + F_{beacon}} \quad (7)$$

where $F_{routing}$ is the frequency of messages for routing, and F_{beacon} is the frequency of beacon messages.

We study all possible combinations of w_1, w_2, $Trust_{beacon}$, and $Trust_{routing}$ and observe that a trust factor with a greater weight can hide the malicious property presented by the other trust factor in the following two circumstances:

1) $Trust_{beacon} < 0.5$ and $w_1 \geq w_2$: The packet routing rate of one-hop RSU is greater than or equal to the beacon message generation rate, and the one-hop RSU behaves maliciously for beacon messages. In this case, $Trust_{routing}$ hides the effect of $Trust_{beacon}$.

2) $Trust_{routing} < 0.5$ and $w_2 \geq w_1$: $Trust_{routing}$ indicates that the one-hop RSU has dropped more than 50% (Xia *et al.*, 2018) of the packets or it has modified the messages before routing. $w_2 \geq w_1$ indicates that the beacon message generation rate of one-hop RSU is greater than or equal to the packet routing rate. Thus, $Trust_{beacon}$ hides the packet drop/modify attribute of one-hop RSU.

To handle the above-mentioned cases, we cut off the weight from a trust factor which hides the malicious activities displayed by other trust factor as follows (Wang *et al.*, 2020):

$$w = aTe^{-(bT)} \quad (8)$$

where T is the pre-defined time interval, $b = w_1$ if $w_1 \geq w_2$ else w_2, and $a = 1 - b$. We assign the new weight w to the trust factor that hides the malicious activities of another trust factor, and $1 - w$ is assigned to the remaining trust factor.

For the remaining combination of w_1, w_2, $Trust_{routing}$, and $Trust_{beacon}$, a trust factor with higher weight does not hide the malicious behavior presented by the other trust factor. Hence, w_1 (computed by Eq. 6) and w_2 (computed by Eq. 7) are not updated following Eq. 8 in these cases.

G. ***Indirect Trust Calculation***

We use the Q-learning method (Guleng *et al.*, 2019) to compute indirect trust. In this technique, every RSU maintains a Q-table containing an entry for every other RSUs and

broadcasts the table with *hello* messages. In the Q-table, each entry contains [$Q(m, n)$, X], where $Q(m, n)$ is the trust value of RSU_m determined by RSU_n, and X is a boolean variable which indicates whether RSU_m is a one-hop neighbor of RSU_n or not. The initial Q-value of each RSU is 0. Suppose RSU_p updates the Q-table and the Q-value of RSU_m at RSU_p is updated as follows:

$$Q_{new}(m, p) = \alpha \times Q_{old}(m, p) \times \{r + \gamma \times avg_{v \in NB_m} Q(m, v)\} + (1 - \alpha) \times Q_{old}(m, p) \quad (9)$$

where $Q_{new}(m, p)$ is the new trust value of RSU_m evaluated by RSU_p, $Q_{old}(m, p)$ is the previously assigned trust value, NB_m is the set of one-hop neighbor RSUs of RSU_m, α is the learning rate set to 0.7 (Guleng *et al*., 2019), $r = Trust_{direct}$ if RSU_m is a one-hop neighbor of RSU_p; otherwise, $r = 0$, and γ is the discount factor set to 0.9 (Guleng *et al*., 2019). Eq. 9 takes the average of trusts provided by the neighbor RSUs to handle the Q-table modification by malicious RSUs and it is denoted as $avg_{v \in NB_m} Q(m, v)$.

Significant communication overhead occurs due to huge message passing to update the entries of Q-table for each RSU in the network. To minimize this communication overhead, we limit both the Q-table size and the number of times the Q-table should be broadcasted to 6 following the hop count constraint discussed in section III-A.

H. Threshold Calculation and Malicious RSU Detection

Each RSU exhibits different behavior from the others. Therefore, it is crucial to maintain an individual threshold value for every RSU. The proposed scheme uses the threshold adjustment mechanism (Kerrache *et al.*, 2018) to identify malicious RSUs where the initial trust value and threshold value for each RSU are set to 0.5. The threshold value varies in the range of [0.5, 1], and the trust value varies in the range of [0, 1]. The threshold value is adjusted according to the changes in trust value which reflects the behavior changes of an RSU. The new threshold value, TH_{new} is determined as follows:

$$TH_{new} = \begin{cases} \beta + 0.5 & if\ \beta > 0 \\ TH_{old} & if\ \beta = 0 \\ 0.5 & if\ \beta < 0 \end{cases} \quad (10)$$

where β is expressed as $\beta = Trust_{old} - Trust_{new}$.

Each RSU computes trust values of other RSUs in the network and compares the trust value with the corresponding threshold value to identify them as *legitimate* or *compromised* RSU. An RSU is classified as *legitimate* when trust value is higher than TH_{new}. Otherwise, it is a *compromised* RSU.

I. Handling Malicious Vehicles

The proposed scheme uses intermediary vehicles between the one-hop neighbor RSUs to verify beacon content and traffic alerts. Although it is not possible to completely overcome the impact of malicious vehicles, we incorporate a mechanism to minimize it. A malicious vehicle in the transmission path can create a false traffic alert or drop or modify the correct alert messages. In our scheme, nonmalicious neighbor vehicles observing the same event verify the traffic alert's authenticity and drop the message if it is incorrect. They share the message with the vehicles within their transmission range, only if it is valid. Vehicles not observing the event rely on the majority opinions of neighbor vehicles and source RSU and verify the alert accordingly. To reduce the effect of malicious vehicles further, the RSU prefers the opinion of the majority of its adjacent vehicles regarding any event message. Section III-D and III-E discuss the handling of rogue vehicles in details for verification of beacon messages and traffic alerts, respectively.

IV. Simulation Results

In this section, we present the results of different experiments conducted to evaluate the performance of the proposed scheme. Besides, the experimental outcomes showing the impact of the proposed method on the current VANET routing protocols are also presented in this section.

A. Experimental Setup

We used Network Simulator 3 (NS-3) (Riley and Henderson, 2010) to evaluate the performance of our proposed scheme. Besides, we used Simulation of Urban Mobility (SUMO) (Behrisch *et al.*, 2011) to construct a realistic vehicular mobility model. A visual network editor of SUMO known as NetEdit (Behrisch *et al,*, 2011) was used to insert and manually position static RSU nodes at every intersection in the traffic environment. After that, vehicles were loaded into this network model. SUMO configuration files were used to generate trace files containing the information of vehicle movements, and they were fed into the NS-3 simulation. The ns2-mobility-helper (ns-3 documentation, 2025) tool in NS-3 was then used to parse this trace file and get the coordinates and velocity of the vehicles. It then assigned the corresponding mobility patterns to network nodes by maintaining temporal and spatial alignment with communication layers. Figure 5 presents our simulation model generated by SUMO.

For simulation, we considered an area of $14km \times 14km$ where 25 RSUs were serving 500 vehicles running with an average speed of $20m/s$. NS-3 uses IEEE 802.11p

communication protocol, which sets the transmission range of vehicles and RSUs to 250m and 900m, respectively. A summary of the simulation parameters is presented in Table I.

We considered the impact of both malicious vehicles and malicious RSUs to evaluate the performance, as the verification of both beacon messages and traffic alerts involves vehicles running on the road. Experiments were conducted with an increasing percentage of malicious RSUs, MR (20%, 40%, and 60%) and an increasing percentage of malicious

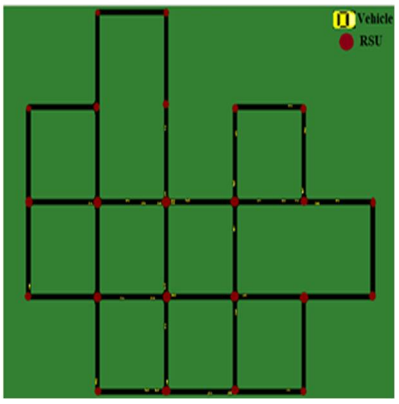


Fig. 5. Simulation traffic model.

Table 1. Parameters used in simulation.

Parameter	Value
Simulation area	14 km × 14 km (Mershad *et al.*, 2012)
Number of vehicles	500 (Jindal and Bedi, 2017)
Transmission range of vehicle	250m (Kerrache *et al.*, 2018)
Vehicle speed	20m/s (Mershad *et al.*, 2012)
Number of RSUs	25 (Mershad *et al.*, 2012)
Distance between RSUs	900m (Bhoi, and Khilar, 2014)
Transmission range of RSU	900m (Bhoi, and Khilar, 2014)
Simulation time	20m

vehicles, MV (5%, 10%, 15%, and 20%). Each experiment ran the simulation for 20 minutes and the results were averaged over 10 iterations with error bars indicate 95% confidence intervals (Guleng *et al.*, 2019). We follow (Guleng *et al.*, 2019; Xia *et al.*, 2019) to adopt an attack model for our proposed scheme. In our experiments, malicious RSUs dropped and forwarded packets with a probability of 0.5. During packet transmission, they modified packets with a probability of 0.5. Rogue RSUs also created flooding attacks and altered information in beacon messages with a probability of 0.5. Further, they altered traffic alerts with a probability of 0.5. On the other hand, malicious vehicles generated *IGNORE_RSU* alert for an accurate beacon message with a probability of 0.5. They dropped *IGNORE_RSU* event or modified the event value with a probability of 0.5. Besides, they modified the traffic alerts with a probability of 0.5.

In our simulation, Q-learning was employed to compute indirect trust of nonneighbor RSUs. The state represents the current trust knowledge an RSU has about nonneighbor RSUs, incorporating Q-value and the trust feedback received from the neighbor RSUs. The action is the periodic update of the Q-value using the defined update formula in Section III-G. The reward was set to zero for nonneighbors due to the absence of direct

trust information. We used a fixed learning rate (α = 0.7) and a discount factor (γ = 0.9) (Guleng *et al.*, 2019).

B. Performance Metrics

We evaluated the performance of the proposed scheme using five performance metrics. Table II enlists the parameters used to define these performance metrics.

Table 2. Parameters used in performance metrics.

Parameter	Description
True Positive (TP)	No. of malicious RSUs identified correctly.
False Positive (FP)	No. of legitimate RSUs identified as malicious RSUs.
True Negative (TN)	No. of legitimate RSUs identified correctly.
False Negative (FN)	No. of malicious RSUs identified as legitimate RSUs.

1) False Positive Rate (FPR): It shows the probability of legitimate RSUs to be identified as malicious RSUs as $False\ Positive\ Rate = \frac{FP}{FP + TN}$ (11)

2) False Negative Rate (FNR): It shows the probability of malicious RSUs to be identified as legitimate RSUs as $False\ Negative\ Rate = \frac{FN}{FN + TP}$ (12)

3) Precision: It is the ratio of correctly detected malicious RSUs to the total number of RSUs that are identified as malicious and defined as $Precision = \frac{TP}{TP + FP}$ (13)

4)Recall: It is the ratio of correctly identified malicious RSUs to the total number of actual malicious RSUs and defined as $Recall = \frac{TP}{TP + FN}$ (14)

5)Accuracy: It is the ratio of correctly identified malicious RSUs and legal RSUs to the total number of RSUs and defined as $Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$ (15)

C. Performance Analysis

In this section, we present our findings on the performance of the proposed scheme derived from the analysis of different experiments. Although the work of (Abhishek *et al.*, 2019) calculates the trust of RSUs, it is not possible to compare this scheme with our proposed method in a meaningful way as each scheme calculates the trust values of RSUs considering the behavior of RSUs in diverse communication scenarios. Hence, we present the results for the proposed system only.

1) ***False Positive Rate (FPR):*** Figure 6a shows that FPR increases with MV and MR and

it reaches to approximately 30% when *MR*=60% and *MV* =20%. Two reasons are mainly working behind the generation of FPR in our scheme. If *MR* increases significantly, all the adjacent RSUs of a legal RSU may behave maliciously, and they can bound the legitimate RSU to drop packets by creating beacon flooding attacks. Thus, a legal RSU is identified as a malicious RSU. Besides, vehicles propagate traffic alerts to help verification of beacon content and alert messages. Rogue vehicles may generate *IGNORE _RSU* alerts for an honest RSU to prove it malicious. Similarly, when an honest RSU reports an event, hostile vehicles can drop or alter the traffic alerts generated by the honest vehicles for the same event and establish the RSU as a malicious one.

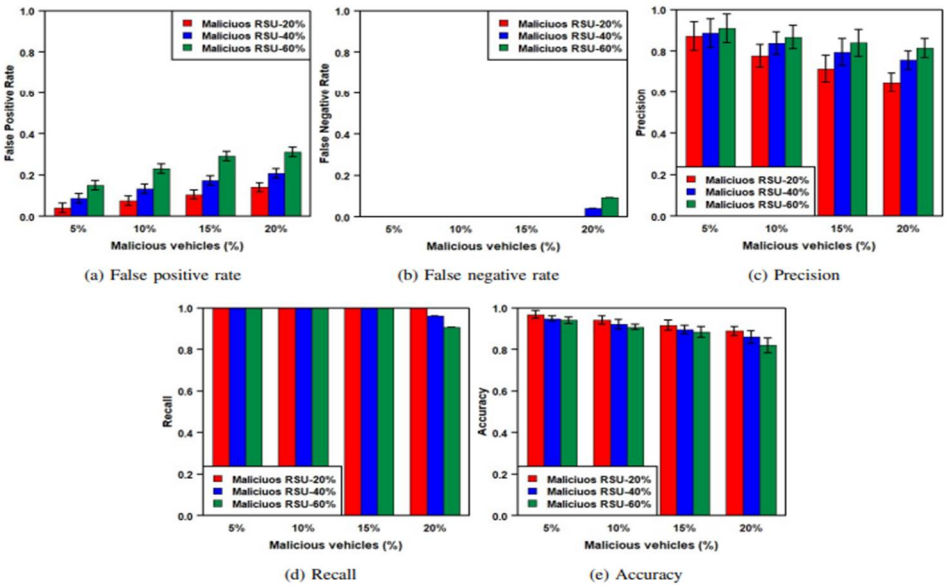


Fig. 6. Performance analysis of the proposed scheme.

2) False Negative Rate (FNR): As shown in Fig. 6b, malicious vehicles have very little influence on the FNR. When a malicious RSU generates a false beacon message, the rogue vehicles can either stop propagating *IGNORE_RSU* or modify the event value. Hence, the malicious attributes of an RSU remain unknown, and it is considered a legitimate RSU. When *MV* is relatively low, the nonmalicious vehicles suppress malicious vehicles' effect by verifying the same beacon messages. Therefore, FNR is not visible up to *MV* =15% in Fig. 6b. In our simulation, a situation where modified traffic alerts by malicious RSUs match with the rogue vehicles' opinion does not occur because of the low frequency of traffic alerts and independent decision-making of malicious vehicles without considering the action of malicious RSUs. Therefore, FNR is not

generated for traffic alerts. Malicious RSUs have no impact on the FNR. The proposed scheme generates maximum 8% FNR for *MR*=60% and *MV* =20% shown in Fig. 6b.

3) Precision: Figure 6c shows that the precision decreases with the rising values of *MV*. As we described in Section IV-C1, the impact of *MV* increases FPR. Thus, the precision decreases with the increasing values of *MV*. For a fixed *MV*, higher values of *MR* increase both TP and FP. Hence, precision increases with *MR* for a fixed *MV*. The precision reaches almost 81% when *MR*=60% and *MV* =20% shown in Fig. 6c. When *MR*=60%, the precision drops approximately 10% at *MV* =20% compared with *MV* =5%. Similarly, the precision drops approximately 26% at *MV* =20% compared with *MV* =5% when *MR*=20%. These results indicate that when *MR* is higher, precision mainly depends on the activities of rogue RSUs. On the other hand, for lower values of *MR*, the precision values are dominated by rogue vehicles' malicious activities.

4) ***Recall:*** The proposed solution identifies nearly all the malicious RSUs shown in Fig. 6d. The recall value is around 92% at *MV* =20%, and *MR*=60%. The recall ratio also indicates that the proposed solution is sensitive to the increasing *MV*. The higher values of *MV* enable the rogue RSUs to hide their malicious properties, as discussed in Section IV-C2. Fig. 6b shows that FNR is visible for higher values of *MV*. Therefore, slight increase of FNR at *MV* =20% in Fig. 6b reduces the recall values at *MV* =20% for *MR*=40%, and *MR*=60% shown in Fig. 6d.

5) ***Accuracy:*** Figure 6e shows the accuracy of the proposed scheme. If *MV* is fixed, accuracy decreases with increasing *MR*. As FP is nearly the same, and FN is rarely visible for fixed *MV*, accuracy depends on TP and TN. Higher values of *MR* increase both TP and FP, reducing FN and TN, respectively. As a consequence, accuracy decreases with higher values of *MR* for a specific *MV*. On the other hand, for fixed *MR*, with higher *MV*, both FP and FN increase, decreasing TN and TP, respectively. Hence, the accuracy decreases gradually with increasing *MV* for a particular *MR*. The proposed scheme achieves an accuracy of approximately 86% when *MR*=60% and *MV* =20%.

D. Impact of Beacon Sensor Data Verification

The proposed scheme considers an RSU's tendency to cause flooding attacks and alter beacon content in computing $Trust_{beacon}$. It incorporates a mechanism to check the correctness of sensor data in beacon messages (Section III-D2) in addition to checking the correctness of speed and density information found in existing literature (Al-Otaibi *et al.*, 2019; Arshad *et al.*, 2018). We conducted a simulation study to evaluate the effect of the sensor data verification mechanism. Experiments were conducted for a fixed number

of malicious vehicles *MV* = 20% and varying numbers of malicious RSUs, *MR*=20%, 40%, and 60%. We collected experimental results for the proposed scheme with (scenario 1) and without (scenario 2) the sensor data verification mechanism and compared the performance in both scenarios. Figure 7 shows the result of comparison which demonstrates a visible reduction in accuracy for both scenarios with the increasing number of malicious RSUs. However, integration of the sensor data verification mechanism improves accuracy, and the highest improvement is observed when *MR* = 60%, yielding an approximate 8% increase in accuracy.

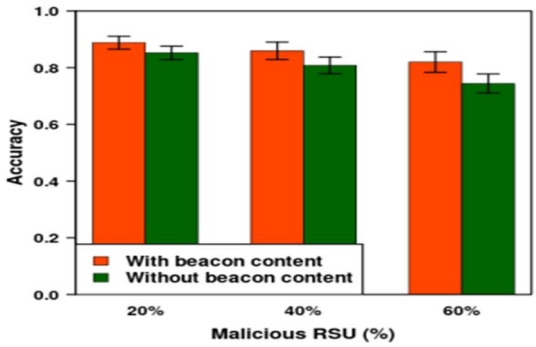


Fig. 7. Impact of sensor data verification in decision accuracy for *MV* =20%.

E. Network Performance Analysis

We incorporated our proposed scheme with several VANET routing protocols such as *Ad Hoc On-Demand Distance Vector (AODV) (*Sallam and Mahmoud, 2015), *Optimized Link State Routing (OLSR)* (Chouhan and Deshmukh, 2015), and *Destination Sequenced Distance Vector (DSDV)* (Rani *et al.*, 2011) to analyze the network performance. As an entity of VANET, RSU also uses these protocols for *R2R* and *V2R* communication (Chouhan and Deshmukh, 2015). We used the same simulation traffic model and parameters as discussed in Section IV-A. To generate data packets, we used 50 vehicles and all of the 25 RSUs as source nodes and considered all entities in the traffic model as receiver nodes. The simulation was performed for varying numbers of malicious RSUs, *MR* (20%, 40% and 60%) and a fixed number of malicious vehicles, *MV* =20%.

1) ***Network Performance Metrics:*** To analyze the network performance we used the following metrics:

1) *Packet Delivery Ratio (PDR):* It is the ratio of the total number of data packets received by destination nodes to the total number of packets sent from source nodes.

$$PDR = \frac{Total\ no.of\ packets\ received}{Total\ no.of\ packets\ sent} \quad (16)$$

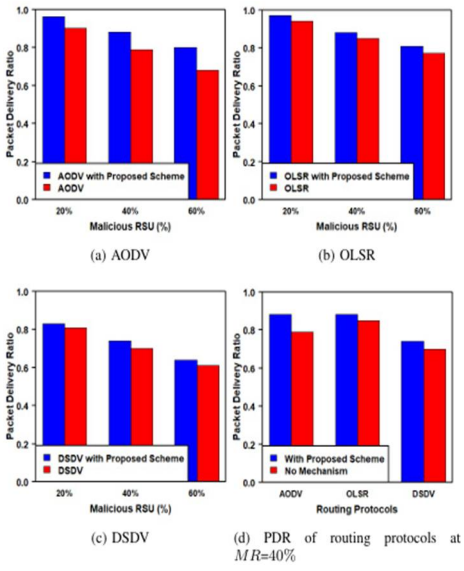
2) *Throughput (Tp):* It is the number of data packets transmitted successfully at a given time.

$$T_p = \frac{Total\ no.\ of\ packets\ transmitted\ successfully}{Total\ time} \quad (17)$$

3) *Average End-to-End (AE2E) Delay:* It is the ratio of the time required to send data packets from source to destination to the total number of packets received.

$$AE2E\ Delay = \frac{\sum(Time\ to\ receive\ -\ Time\ to\ send)}{Total\ no.of\ packets\ received} \quad (18)$$

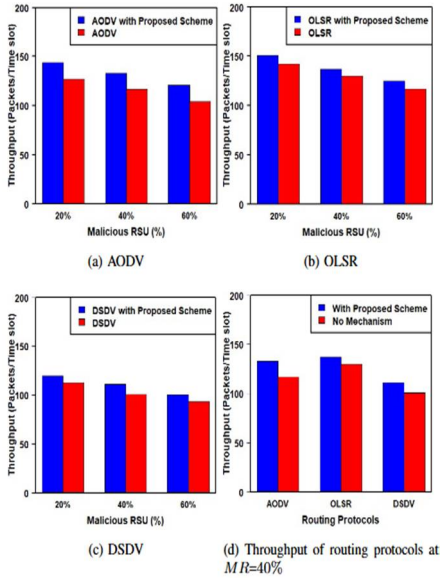
In the subsequent sections, we analyzed the performance of the network considering the presence and absence of the proposed trust model in all the routing protocols mentioned in Section IV-E.



(a) AODV (b) OLSR

(c) DSDV (d) PDR of routing protocols at MR=40%

Fig. 8. Packet delivery ratio on various values of *MR* for *MV* =20%.



(a) AODV (b) OLSR

(c) DSDV (d) Throughput of routing protocols at MR=40%

Fig. 9. Throughput on various values of *MR* for *MV* =20%.

2) ***Packet Delivery Ratio (PDR):*** In the fundamental AODV, OLSR and DSDV protocols where the malicious RSU detection mechanism is missing, an RSU forwards packets to next-hop RSU without considering the malicious behavior of that RSU. If the next-hop RSU is malicious, it can drop packets, resulting in low PDR. In case of packet drops, packet retransmissions take place in each protocol. However, integration of the proposed trust model improves the PDR in all the protocols as an RSU can decide to exclude one-hop malicious RSUs from packet forwarding. As shown in Fig. 8a, 8b and 8c, the average improvement for AODV, OLSR, and DSDV is approximately 12%, 4% and 4%, respectively. Nevertheless, PDR decreases with the increasing number of malicious RSUs as they can jam the network through excessive beacon broadcasting or increase packet drop. From Fig. 8d it is clear that the basic OLSR protocol performs better than the other fundamental protocols. If any disconnection occurs, OLSR finds a new route faster than other protocols using routing tables. In contrast, DSDV takes a longer time to find a new route and, therefore, results in low PDR. In case of AODV, it is not facilitated like OLSR to get route information from some selected nodes known as MultiPoint Relay (MPR). Hence, AODV has lower PDR compared to OLSR. On the other hand, incorporation of the proposed trust model results in similar PDR for both AODV and OLSR as AODV usually creates a route immediately if needed, whereas OLSR updates the routing table periodically. During route discovery, AODV also takes the advantages of the trust model to avoid malicious RSUs. Hence, PDR increases significantly for AODV.

3) ***Throughput (Tp):*** Figure 9 presents the throughput of each routing protocol with/without the proposed trust model. As discussed earlier, each routing protocol with the trust model improves the packet delivery ratio and as a result the throughput increases. As shown in Fig. 9a, 9b and 9c, the average improvement for *AODV*, *OLSR*, and *DSDV* is approximately 14%, 6%, and, 8%, respectively. Similar to the *PDR*, we observe from Fig. 9d that the *OLSR* protocol exhibits best throughput, which is followed by the performance of *AODV* and *DSDV* protocols, respectively due to their underlying mechanism as mentioned in Section IV-E2.

4) ***Average End-to-End (AE2E) Delay:*** When the proposed trust model merges with the routing protocols, they select honest next-hop RSU to propagate messages. Hence, packet drops are reduced for each protocol which ultimately reduces packet retransmissions. As a consequence, end-to-end delay decreases as shown in Fig. 10. However, it is observed that the improvement in the end-to-end delay is minimal. Our proposed model only detects the malicious RSUs and does not exclude them from the network. Therefore, malicious impact such as beacon flooding remains in the network that can cause

congestion. Though end-to-end delay for both *OLSR* and *AODV* is nearly same as shown in Fig. 10d, *OLSR* shows slightly better performance due to the routing efficiency. Once again, *DSDV* shows the worst end-to-end delay performance.

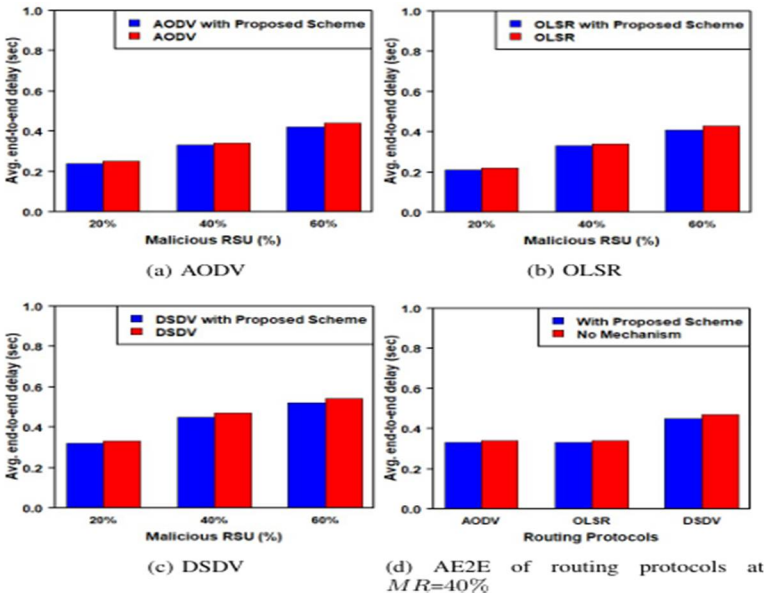


Fig. 10. Average end-to-end delay on various values of *MR* for *MV* =20%.

V. Conclusion and Future Work

In this paper, we proposed a trust-based malicious RSU detection mechanism for an edge-enabled VANET. Our proposed scheme analyzes R2R communication patterns to find the deviation in RSUs' behavior and assigns trust scores accordingly to distinguish malicious RSUs from nonmalicious ones. Besides, we proposed a mechanism to evaluate trust values based on the correctness of the beacon content provided by an RSU. The simulation results reveal that our scheme detects approximately 92% malicious RSUs and decides the type of RSUs with an accuracy of nearly 86% in the presence of rogue vehicles. Besides, the proposed scheme contributes a moderate network performance improvement of 14% when incorporated with the *AODV* routing protocol. In the future, we aim to include a sophisticated mechanism to minimize the impact of malicious vehicles. We also plan to utilize different machine learning mechanisms to identify malicious RSUs and want to study the performance of the proposed scheme using real-world data sets.

References

Abhishek, N. V., T. J. Lim, B. Sikdar and B. Liang. 2019. Detecting RSU misbehavior in vehicular edge computing. In: *Proc. IEEE/CIC Int. Conf. Commun.* pp. 42-47.

Abhishek, N. V. and T. J. Lim. 2022. Trust-based adversary detection in edge computing assisted vehicular networks. *J. Commun. Netw.* **24**(4): 451-462.

Ahmed, S., M. U. Rehman, A. Ishtiaq, S. Khan, A. Ali, and S. Begum. 2018. VANSec: Attack-resistant VANET security algorithm in terms of trust computation error and normalized routing overhead. *J. Sens.* **2018**(1): 6576841.

Alnasser, A., and H. Sun. 2021. Trust-based model for securing vehicular networks against RSU attacks, in *Proc. IEEE INFOCOM Conf. Comput. Commun. Workshops*. IEEE, pp. 1-6.

Al-Otaibi, B., N. Al-Nabhan, and Y. Tian. 2019. Privacy-preserving vehicular rogue node detection scheme for fog computing. *Sensors*, **19**(4): 965.

Arshad, M., Z. Ullah, M. Khalid, N. Ahmad, W. Khalid, D. Shahwar and Y. Cao. 2018. Beacon trust management system and fake data detection in vehicular ad-hoc networks. *IET Intell. Transport Syst.*, **13**(5): 780-788.

Behrisch, M., L. Bieker, J. Erdmann, and D. Krajzewicz. 2011. SUMO–simulation of urban mobility: An overview. In: *Proc. 3rd Int. Conf. Adv. Syst. Simul.*

Bhoi, S. K. and P. M. Khilar. 2014. IJS: An intelligent junction selection based routing protocol for VANET to support ITS services. *Int. Scholarly Res. Notices.* **2014**: 1-15.

Chouhan, T.S. and R.S. Deshmukh. 2015. Analysis of DSDV, OLSR and AODV routing protocols in VANETS scenario: using NS3. In: *Proc. Int. Conf. Comput. Intell. Commun. Netw.* IEEE, pp. 85-89.

Guleng, S., C. Wu, X. Chen, X. Wang, T. Yoshinaga and Y. Ji. 2019. Decentralized trust evaluation in vehicular internet of things. *IEEE Access*. **7**: 15980-15988.

Hao, Y., Y. Cheng and K. Ren. 2008. Distributed key management with protection against RSU compromise in group signature based VANETs. in *Proc. IEEE Globecom*, pp. 1-5.

Huang, L., H. Jiang, Z. Zhang, Z. Yan and H. Guo. 2017. Efficient data traffic forwarding for infrastructure-to-infrastructure communications in VANETs. *IEEE Trans. Intell. Transp. Syst.* **19**(3): 839-853.

Hussain, R., J. Lee and S. Zeadally. 2020. Trust in VANET: A survey of current solutions and future research opportunities. *IEEE Trans. Intell. Transp. Syst.* **22**(5): 2553-2571.

Jalooli, A., H. Khan and L. Purohit. 2024. Blockchain-enabled collaborative forged message detection in RSU-based VANETs. In *Proc. 8th Cyber Sec. Netw. Conf. (CSNet)*, pp. 60-67.

Jin, H., and P. Papadimitratos. 2018. Expedited beacon verification for VANET. In: *Proc. 11th ACM Conf. Secur. Privacy Wirel. Mob. Netw.* pp. 283-284.

Jindal, V., and P. Bedi. 2017. Reducing waiting time with parallel preemptive algorithm in VANETs. *Veh. Commun.* 7: 58-65.

Kerrache, C.A., A. Lakas, N. Lagraa and E. Barka.2018. UAV-assisted technique for the detection of malicious and selfish nodes in VANETs. *Veh. Commun.* **11**: 1-11.

Lee, E., E.-K. Lee, M. Gerla and S. Y. Oh. 2014. Vehicular cloud networking: architecture and design principles. *IEEE Commun. Mag.* **52**(2): 148-155.

Liu, J., W. Yang, J. Zhang and C. Yang. 2020. “Detecting false messages in vehicular ad hoc networks based on a traffic flow model. *Int. J. Distrib. Sens. Netw.* **16**(2): 1-12.

Lone, F.R. and H.K. Verma. 2025. MP-TMD: A multidimensional plausibility-driven cooperative trust model for multiple misbehaviour detection in intelligent transportation systems. *Cluster Computing*, **28**(3): 181.

Lu, X., X. Wan, L. Xiao, Y. Tang and W. Zhuang. 2018. Learning-based rogue edge detection in VANETs with ambient radio signals. In: *Proc. IEEE Int. Conf. Commun.*, pp. 1-6.

Maglaras, L. A., P. Basaras, and D. Katsaros. 2013. Exploiting vehicular communications for reducing CO_2 emissions in urban environments. In: *Proc. Int. Conf. Connected Veh. Expo (ICCVE).*, pp. 32-37.

Malik, N., P. Nanda, A. Arora, X. He and D. Puthal. 2018. Blockchain based secured identity authentication and expeditious revocation framework for vehicular networks. In: *Proc. 17th Int. Conf. Trust, Secur. Privacy Comput. Commun. (TrustCom)*. IEEE, pp. 674-679.

Marti, S., T. J. Giuli, K. Lai and M. Baker.2000. Mitigating routing misbehavior in mobile ad hoc networks. In: *Proc. 6th Annu. Int. Conf. Mob. Comput. Netw.* pp. 255-265.

Mershad, K., H. Artail and M. Gerla. 2012. ROAMER: Roadside units as message routers in VANETs. *Ad Hoc Netw.* **10**(3): 479-496.

ns-3 documentation. 2025. ns3::Ns2MobilityHelper Class Reference. Accessed June 6, 2022. [Online]. Available: https://www.nsnam.org/docs/release/3.19/doxygen/classns3_1_1_ns2_mobility_helper.html

Onieva, J.A., R. Rios, R. Roman and J. Lopez. 2019. "Edge-assisted vehicular networks security," *IEEE Internet Things J.* **6**(5): 8038-8045.

Paranjothi, A., M. Atiquzzaman and M.S. Khan. 2020. F-RouND: Fog-based rogue nodes detection in vehicular ad hoc networks. In: *Proc. IEEE Globecom*, pp. 1-6.

Patil, R., and M. P. Tahiliani. 2014. Detecting packet modification attack by misbehaving router. In: *Proc. 1st Int. Conf. Netw. Soft Comput.* pp. 113-118.

Qu, F., Z. Wu, F.-Y. Wang and W. Cho. 2015. A security and privacy review of VANETs. *IEEE Trans. Intell. Transp. Syst.* **16**(6): 2985-2996.

Rani, P., N. Sharma and P.K. Singh. 2011. Performance comparison of VANET routing protocols. In *Proc. 7th Int. Conf. Wirel. Commun. Netw. Mob. Comput.* IEEE, pp. 1-4.

Riley, G.F. and T. R. Henderson. 2010. The ns-3 network simulator. In: *Model. Tools Netw. Simul.* Berlin, Heidelberg: Springer, pp. 15-34.

Santa, J., F. Pereñíguez, A. Moragón and A. F. Skarmeta. 2014. Experimental evaluation of CAM and DENM messaging services in vehicular communications. *Transp. Res. Part C: Emerg. Technol.* **46**: 98-120.

Sajjad, S.M., S.H. Bouk and M. Yousaf. 2015. Neighbor node trust based intrusion detection system for WSN. *Procedia Comput. Sci.* **63**: 183-188.

Sallam, G. and A. Mahmoud. 2015. Performance evaluation of OLSR and AODV in VANET cloud computing using fading model with SUMO and NS3. In: *Proc. Int. Conf. Cloud Comput.* IEEE, pp. 1-5.

Sheikh, M.S., J. Liang and W. Wang. 2019. A survey of security services, attacks, and applications for vehicular ad hoc networks (VANETs). *Sensors* **19**(16): 3589.

Soleymani, S.A., A.H. Abdullah, W.H. Hassan, M.H. Anisi, S. Goudarzi, M.A.R. Baee and S. Mandala.2015. Trust management in vehicular ad hoc network: a systematic review. *EURASIP J. Wirel. Commun. Netw.* **2015**(1): 1-22.

Tripathi, K.N. and S. Sharma. 2019. A trust based model (TBM) to detect rogue nodes in vehicular ad-hoc networks (VANETS). *Int. J. Syst. Assur. Eng. Manag.* pp. 1-15.

Van der Heijden, R.W., S. Dietzel, T. Leinmüller and F. Kargl. 2018. Survey on misbehavior detection in cooperative intelligent transportation systems. *IEEE Commun. Surveys Tuts.* **21**(1): 779-811.

Wang, T., G. Zhang, M.Z.A. Bhuiyan, A. Liu, W. Jia and M. Xie. 2020. A novel trust mechanism based on fog computing in sensor–cloud system. *Future Gener. Comput. Syst.* **109**: 573-582.

Xia, H., S.-s. Zhang, B.-x. Li, L. Li and X.-g. Cheng. 2018. Towards a novel trust-based multicast routing for VANETs. *Secur. Commun. Netw.* **2018**(1): 7608198.

Xia, H., S.-s. Zhang, Y. Li, Z.-k. Pan, X. Peng and X.-z. Cheng. 2019. An attack-resistant trust inference model for securing routing in vehicular ad hoc networks. *IEEE Trans. Veh. Technol.* **68**(7): 7108-7120.

Yao, Y., X. Chang, J. Mišic´, V. B. Mišic´ and L. Li. 2019. BLA: Blockchain-assisted lightweight anonymous authentication for distributed vehicular fog services. *IEEE Internet Things J.* **6**(2): 3775-3784.

Zaidi, K., M. Milojevic, V. Rakocevic, and M. Rajarajan. 2014. Data-centric rogue node detection in VANETs. In: *Proc. IEEE 13th Int. Conf. Trust, Secur. Privacy Comput. Commun.* pp. 398-405.

Zaidi, K., M.B. Milojevic, V. Rakocevic, A. Nallanathan and M. Rajarajan. 2015. Host-based intrusion detection for VANETs: A statistical approach to rogue node detection. *IEEE Trans. Veh. Technol.* **65**(8): 6703-6714.

Zhang, X. and X. Chen. 2019. Data security sharing and storage based on a consortium blockchain in a vehicular ad-hoc network. *IEEE Access*. 7: 58241-58254.

Zhang, Y., C. Cheong, S. Li, Y. Cao, X. Zhang and D. Liu. 2024. False message detection in Internet of Vehicle through machine learning and vehicle consensus. *Inf. Process. Manage.* **61**(6): 103827.

(Revised copy received on 24/06/2025)