Secured Traffic Monitoring in VANET

Ayan Roy
Department of Computer Science
Missouri University of Science and Technology, USA
ar3g3@mst.edu

Sanjay Madria
Department of Computer Science
Missouri University of Science and Technology, USA
madrias@mst.edu

Abstract—Vehicular Ad hoc Networks (VANETs) facilitate vehicles to wirelessly communicate with neighboring vehicles as well as with roadside units (RSUs). However, the existence of inaccurate information within the network can cause traffic aberrations and also disrupt the normal functioning of any traffic monitoring system. Thus, determining the credibility of broadcast messages originating from the region of interest (ROI) is crucial under a malicious environment. Additionally, a breach of privacy involving a vehicle’s private information, such as location and velocity, can lead to severe consequences like unauthorized tracking and masquerading attack. Thus, we propose an edge cloud based privacy-preserving secured decision making model that employs a heuristic based on vehicular data such as GPS location and velocity to authenticate traffic-related information from the ROI under different traffic scenarios such as congestion. The effectiveness of the proposed model has been validated using VENTOS, SUMO, and Omnet++ simulators, and also by using a simulated cloud environment. We compare our proposed model to the existing peer-based authentication model, the majority voting model, and the reputation-based system under different attack scenarios. We show that our model is capable of filtering malicious vehicles effectively and provide accurate traffic information under the presence of at least one non-malicious vehicle within the ROI.

Index Terms—Secure, Privacy, Edge, VANET

I. INTRODUCTION

Vehicular Ad hoc Networks (VANETs) allow wireless communication from vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) such as with roadside units (RSU) for better traffic management. Using the dedicated short range communication (DSRC) protocol, every vehicle broadcasts information about traffic events such as accidents, traffic congestion, and traffic violations to nearby vehicles as well as road side infrastructures. However, the presence of malicious vehicles at the region of interest (ROI) can negatively influence the traffic monitoring of the region. Existing strategies such as the peer authentication model [1][2], majority voting [3][1] and the reputation-based system [4] provide traffic monitoring by assuming the concept of adversarial parsimony [3] but it becomes challenging to validate the traffic-related responses when the malicious vehicles are in the majority.

As a motivating example, an inaccurate traffic response originating from the ROI can detour an emergency vehicle from taking the shortest path for evacuation in case of a disaster. Furthermore, an increase in the autonomy of vehicles makes the infrastructure even more vulnerable to security attacks. The remote hacking of a Jeep Cherokee in 2015 [5] highlights the feasibility of such attacks. Additionally, the breach in the privacy of individual vehicle information can lead to dire consequences, such as unauthorized tracking of government officials as well as vehicles identity theft. The purpose of an attacker is to profile a driver’s habits based on GPS location, velocity, acceleration, and the unique ID of the vehicle. Under such constraints, it is necessary to ensure the anonymity and unlinkability of the information shared vehicles. Anonymity ensures that every vehicle remains anonymous while exchanging information whereas, unlinkability ensures the inability to trace the identity of a vehicle based on the information exchanged by it. Furthermore, such a system should ensure conditional privacy, which means that the identity of the attacker can be revealed in case of a conflict.

To resolve the issue, the United States Department of Transport (USDOT) introduced Security Credential Management System (SCMS) [6] that leverages V2V and V2I communication among vehicles, and public key infrastructure (PKI) to ensure message integrity, authenticity, privacy, and interoperability. However, it does not provide any algorithm for misbehavior detection in the network. In other words, if authentic vehicles misbehave and provide inaccurate traffic-related information from the ROI, SCMS does not provide any algorithm in place to validate the information and filter out the malicious vehicles and responses from the network, and thus it obtains inaccurate traffic information. This problem is elevated even more if the malicious vehicles at the ROI form the majority and try to disrupt the traffic monitoring system. To address these shortcomings, we propose an edge cloud based secured and privacy preserving decision making model by leveraging the PKI infrastructure to validate traffic-related information from the ROI and filter malicious vehicles, even if they form the majority of vehicles within the ROI. Each edge server is associated with a different region and is connected to a centralized server. By leveraging the concept of the edge server, the decision making model is brought closer to the ROI concerned, which reduces the latency in the decision and reduces the bottleneck on the centralized server. Our aim is to design and develop a heuristic by leveraging the reported vehicular information such as GPS location, velocity and authentic vehicle ID to validate the traffic-related information from the ROI by preserving the privacy of information of the individual vehicles from other nearby vehicles. The main contributions of the proposed work are as follows:

- Develop a privacy preserving, secured heuristic solution
that overcomes the shortcomings of the current state-of-the-art models and validates the traffic-related information from the ROI using the recorded GPS location of each vehicle, the vehicles’ velocity and encrypted neighboring vehicles’ IDs under the influence of at least one non-malicious vehicle within the ROI, unlike the assumption of the majority of non-malicious vehicles considered in the state-of-the-art models.

- Design a new dynamic data structure called the Decision Similarity Graph based on the vehicle location, and leverage the Point of Conflict concept to filter out malicious vehicles within the ROI using the conflicting event recorded by any 2 neighboring vehicles.
- Validate the effectiveness of the proposed model as compared to the other existing models under different scenarios using simulations and experiments, and show that the proposed model effectively validates traffic-related information and filters the malicious vehicles and their responses from the network. The proposed model has been compared only against the state-of-the-art models that leverage the V2X communication infrastructure to validate the traffic-related information from the ROI.

II. RELATED WORK

Younes et al. in [7] proposed a technique where every registered vehicle at the ROI advertises information about the road condition. Every neighboring vehicle accepts the information only if it reaches above a certain threshold. The vehicle closest to the RSU encrypts the data and sends it to the RSU which determines the status of the traffic. However, the proposed model fails to respond if the vehicle closest to the RSU maliciously drops the information or it accepts and transmits incorrect information to the RSU. A dynamic threshold based peer authentication technique has been proposed in [2] in which every vehicle has a threshold for accepting messages. However, if the threshold for the vehicle is too high or too low, the model fails to perform efficiently. In [8], the authors proposed a concept of reactive group where the vehicles that group decide and agree on an event. The model fails to respond in case the number of malicious vehicles is higher than non-malicious vehicles.

A threshold based peer authentication has been proposed in [9] where the model detects an event that has occurred at the ROI such as congestion or accident. The RSU keeps public-private key pair of each vehicle at the ROI, thus, needs to store a large number of key pairs. It periodically broadcasts the revocation list about the possible malicious vehicles. The monitoring events are broadcasted by the vehicle based on certain criteria. At this stage, a malicious user can purposefully drop a message. Based on the model, if majority of vehicles at the ROI are malicious then it will accept the monitoring event, and record the incorrect data. Hence, in such a scenario, the system fails to provide accurate traffic information.

The authors in [3] proposed a technique that allows nodes to detect incorrect information, and identify the malicious nodes with high probability. However, the proposed model assumes that the majority of the vehicles in the decision making process are non-malicious, and it further assumes that the sensor data is tamper-proof. Since the number of non-malicious vehicles is more, it nullifies the events recorded by the malicious vehicles. However, the method fails when the number of malicious vehicles outnumbers non-malicious vehicles.

III. SYSTEM AND THREAT MODELS AND ASSUMPTIONS

A. System Model

The entities involved in the architecture of our proposed model, represented in Figure 1, have been described below.

- Edge Server: A trusted entity associated with a small region, such as a county, which analyzes traffic conditions or detects events like accidents or congestion. The edge server is responsible for accurate traffic monitoring based on the proposed heuristic, and it filters malicious vehicles. The use of edge server scales the proposed model for large VANETs using the distributed cloud concept.
- Centralized Server: A trusted entity that is responsible for analyzing the traffic scenario of a city or state. Since traffic congestion in a region will have a cascading effect
on other surrounding regions, the decisions from different edge servers deployed in small regions are analyzed by the centralized server in order to generate an overview of a traffic scenario for a large region.

- **Decision Similarity Graph (DSG):** The Decision Similarity Graph (DSG), represented in Figure 2, is utilized by the edge server to filter malicious vehicles within the ROI, which is explained later in this paper. DSG is an ordered pair of the form:

  \[ DSG = (V, E) \]

  where \( V = \{V_1, V_2, V_3, \ldots, V_n\} \) represent the \( n \) vehicles from which the responses are received and \( E = \{E_1, E_2, E_3, \ldots, E_n\} \) are undirected edges that represent the neighborhood between any two vehicles.

![Fig. 2: Decision Similarity Graph](image)

- **AES encryption algorithm** [12]: Every vehicle, \( V_{id} \), at the ROI uses the AES 128 bit symmetric encryption algorithm to generate a unique key, \( Key_i \), which is used to encrypt the vehicular data such as ID, GPS location, and velocity before sending the data to the edge server via the RSU. Since the AES 128 bit algorithm is faster than the AES 192 and 256 bit algorithms [13] respectively, the proposed model reduces the latency due to encryption.

- **Schmidt-Samoa cryptosystem** [14]: Every edge server using the Schmidt-Samoa cryptosystem generates a public key, \( G_{public} \), and its associated private key, \( G_{private} \), which is utilized by the vehicles within the ROI for a secure key, \( Key_i \), exchange as well as for the privacy preserved \( V_{id} \) broadcast to its neighboring vehicles. Edge servers are associated with different regions and each edge server generates its own \( G_{public} \) and \( G_{private} \). We prefer the Schmidt-Samoa cryptosystem over RSA [15] because inverting the RSA function is a hard problem. Also, unlike Rabin [16], this algorithm does not produce any ambiguity in the decryption at the cost of the encryption speed. The Schmidt-Samoa cryptosystem is also preferred over the elliptic-curve cryptosystem [17] because of the implementation issues faced by the latter and also the lack of research on the latter cryptosystem [18].

- **Elgamal Digital Signature Scheme (EGDSS)** [19]: Every vehicle, \( V_{id} \), before entering into a VANET communication, is registered with the trusted centralized server using a private key, \( veh_{private} \), and its associated public key, \( veh_{public} \), which are generated using the EGDSS algorithm. The \( veh_{private} \) is loaded onto the on-board-unit (OBU) of \( V_{id} \) which is used to authenticate its identity to the edge server. Instead of EGDSS, the digital signature algorithm (DSA) [20] could also be used without affecting the model’s effectiveness.

### B. Threat Model

We assume that vehicles can perform the following attacks:

- **Message and GPS information spoofing:** Malicious vehicles can manipulate any recorded event to divert the decision making process. For example a malicious vehicle can record congestion when the road is not congested. Furthermore, they can manipulate the velocity and recorded GPS location as evidence to support its event reporting.

- **Masquerading, Collusion and Sybil attack:** A malicious vehicle may try to conceal its own identity after reporting some other \( V_{id} \), and thus impersonate some other vehicle. It can also send manipulated traffic-related information under different registered \( V_{id} \), thereby performing a sybil attack. Also, a majority of malicious vehicles within the ROI can collude to disrupt the decision making system within the ROI.

- **Conditional Privacy Preservation:** A malicious vehicle can try to snoop into the data packet of other vehicles to obtain private information like the neighboring \( V_{id} \) and the event recorded by it. It may gain access into a neighboring non-malicious vehicle’s \( V_{id} \) for unauthorized tracking as well as gain access to \( Key_i \) generated by the neighboring non-malicious \( V_{id} \) (using AES 128) to violate the integrity of the non-malicious data packet. Thus, the identity of every vehicle should be concealed from other vehicles (anonymity) and should be unlinkable (unlinkability) [21].

- **Message Integrity attack:** A malicious vehicle may intercept a data packet of any non-malicious vehicle, modify the information, and send it to the RSU for decision making.

- **Denial of Service (DoS):** A malicious vehicle may not send its vehicular data to the RSU in an attempt to degrade the decision making and traffic monitoring system.

### C. Assumptions

We assume that non-malicious vehicles always send requested information required in the proposed model to the edge server via the RSU, unlike malicious vehicles, which may refrain from sending required information; every challenge and response packet is sent and received by the RSU as well as by the vehicles. If a packet is dropped unintentionally by a non-malicious vehicle due to a network problem or channel congestion, it is neglected by the proposed model. However, for the efficient working of the proposed model, the edge server must obtain at least one non-malicious response to validate the traffic-related information from the ROI. The centralized server and the edge server are trusted [22] entities, and a majority of vehicles at the ROI participate in decision making. The requesting vehicle is assumed to be non-malicious. All the vehicles are considered to have a unique ID of uniform length that is known by the edge servers. This is assumed to be allocated when a vehicle is licensed or registered with the department of motor vehicles. Every vehicles’ information that
does not reach the edge server is not taken into consideration for the decision making. The edge server must receive at least one conflicting neighbor of any malicious vehicle existing at the ROI. Every vehicle is loaded with its \textit{veh\_private}, that is generated using \textit{EGDSS} before participating in VANET communication. Moreover, the RSUs deployed at the ROI are assumed to be trusted. Finally, in the proposed model we consider the recorded event to be either congested or non-congested. The proposed model is not appropriate for any decision making, such as turning the steering wheel or increasing the acceleration, that requires real time decisions and the potential of failure that may be fatal in many situations. Also in order to encrypt various data in the proposed model, we have used the ASCII format to convert the characters into strings. Here we can also use any other number format for conversion, such as Binary Coded Decimal (BCD), without affecting how the proposed model works.

IV. PROPOSED MODEL

Figure 3 represents an overview of the proposed model. It uses a privacy-preserving heuristic that leverages the GPS location and velocity of the reporting vehicle as well as the encrypted neighboring \textit{Vd}s of the reporting vehicle, i.e., vehicles that are within its transmission range (usually around 600 meters [23]), to validate traffic-related information at the ROI under the presence of at least one non-malicious vehicle. A vehicle requesting traffic-related information from an ROI under the presence of at least one non-malicious vehicle, can request the traffic condition at another ROI under a different edge server. Under one edge server, the requesting vehicle, can request the traffic condition at another ROI via wireless communication. The centralized server is directly associated with all the edge servers.

A. Key Generation and Request Dissipation

On receiving the request from the centralized server, the edge server associated with the ROI generates \textit{Gpublic} and its associated \textit{Gprivate} using equations 1 and 2, respectively, as defined by the Schmidt-Samoa cryptosystem.

\[
\begin{align*}
\text{G}_{\text{public}} & = p^2 \times q \\
\text{G}_{\text{private}} & = \text{G}_{\text{public}}^{-1} \mod (\text{lcm}(p-1,q-1))
\end{align*}
\]

where \( p \) and \( q \) are 2 large prime numbers chosen by the edge server. Furthermore, the edge server broadcasts \textit{Gpublic} within the ROI via the RSUs.

B. Encrypted ID Broadcast and Response Acquisition

According to the proposed heuristic, every vehicle needs to know the encrypted \textit{Vd}s of its neighboring vehicles only when generating the \textit{data\_packet}. This requirement has been justified later in this section. To preserve the privacy in the proposed model, after receiving the \textit{Gpublic}, the vehicle encrypts its \textit{Vd} using \textit{Gpublic} and generates \textit{enc\_id}, which is defined as the encrypted \textit{Vd} as discussed in Algorithm 1 (lines 6-10), and broadcasts it to its nearby vehicles.

\begin{algorithm}[H]
\caption{Generate \textit{enc\_id} and \textit{\tau} for a vehicle}
\begin{algorithmic}[1]
\State \textit{key}[0 - 9, A - Z, a - z] = [00 - 09, 10 - 35, 46 - 71]
\State \textit{Key}_i \leftarrow \text{Key of a vehicle generated using AES 128}
\State \textit{Key}_m \leftarrow \text{Stores the integer value of } \textit{Key}_i \text{ following notations in step 1}
\State \text{count} \leftarrow \text{length\_of(} \textit{Vd}\text{)} - 1
\State \text{Key\_length} \leftarrow \text{length\_of}(\text{Key}_i) - 1
\For {\text{i in range of } 0 - \text{length\_of(} \textit{Vd}\text{)} - 1}
\State \text{message} = \text{message} + \text{count\_ascii\_of(} \textit{Vd}\text{.charAt(i)})
\EndFor
\State \text{count\_ascii\_of(} \textit{Vd}\text{.charAt(i)})
\State \text{end for}
\State \textit{enc\_id} \leftarrow \text{message} \mod (\text{Gpublic)}
\State \text{end for}
\State \textit{Key\_m} \leftarrow \text{Key\_m} \mod (\text{Gpublic)}
\State \textit{\tau} \leftarrow \text{Key\_m} \mod (\text{Gpublic)}
\Return \textit{enc\_id}, \textit{\tau}
\end{algorithmic}
\end{algorithm}

vehicle having received \textit{threshold} number of \textit{enc\_ids} from neighboring vehicles generates \textit{data\_packet} that consists of the vehicle’s information. The significance of \textit{threshold} in the proposed model is to model the size of the \textit{data\_packet} in such a way that it consumes lesser bandwidth to send the vehicular information by the RSU.

\[
data\_packet = < \textit{Vd}, ds(\textit{Vd}), event, vel\_id, GPS\_id, enc\_ids >
\]

where \( ds(\textit{Vd}) \) is defined as \textit{Vd} digitally signed with \textit{veh\_private}, generated using \textit{EGDSS} algorithm for authenticating itself, \textit{enc\_ids} refers to the encrypted ids of the neighboring \textit{Vd}s, \textit{vel\_id} and \textit{GPS\_id} are the velocity and GPS location of the vehicle respectively at the time of generating the \textit{data\_packet}, while the \textit{event} field indicates the event (such as congestion) recorded by the \textit{Vd}. This \textit{event} can be manipulated by a malicious vehicle to report congestion when it is not. We do not preexamine the recorded GPS location of the vehicles and the proposed model does not deal with the precision of the GPS location of a \textit{Vd}, as the GPS location recorded is utilized to filter out malicious vehicles in the heuristic described later. The precision of the GPS is insignificant in the proposed model. Thereafter, every \textit{Vd} at the ROI generates its own symmetric key, \textit{Key}\_i, using the AES 128 bit algorithm and generates \textit{encrypted\_data\_packet} consisting
of every piece of information that needs to be sent to the edge server in the encrypted form.

\[ \text{encrypted}_p = \tau, \text{data}_{\text{packet}}' \]

where \( \tau \) is obtained by encrypting \( K_{\text{v}i} \) of a \( V_{\text{id}} \) with \( G_{\text{public}} \) (Algorithm 1 lines 11-15) and \( \text{data}_{\text{packet}}' \) is obtained by encrypting \( \text{data}_{\text{packet}} \) with \( K_{\text{v}i} \). The purpose of this step is to preserve the privacy and integrity of \( K_{\text{v}i} \) and the \( \text{data}_{\text{packet}} \). This step also facilitates the secure key exchange algorithm, which has been analyzed in the security analysis section. Every \( V_{\text{id}} \) broadcasts the \( \text{encrypted}_p \) received by the nearby RSU. The RSU waits for \( \sigma \) seconds or threshold number of \( \text{encrypted}_p \) before sending them to the edge server. The waiting time, \( \sigma \), ensures that the RSU constrains the time of the proposed model in case the threshold packets take more time, especially in situations where the traffic flow is low. On the other hand, if the traffic flow is high, within \( \sigma \) time, the RSU can receive a large number of \( \text{encrypted}_p \) which can significantly increase the length of the aggregated packet under which it appends threshold \( \text{encrypted}_p \) obtained before \( \sigma \) seconds. Thus, using this trade-off, the RSU generates the \( \text{aggregated}_p \) and sends it to the edge server.

\[ \text{aggregated}_p = < \text{rsu}_i, \text{location}, \text{encrypted}_p > \]

where \( \text{rsu}_i \) and \( \text{location} \) are respectively the unique \( \text{id} \) and location of the RSU sending the packets to the edge server. The use of this information for the decision making process by the edge server is explained in the subsequent subsections in this paper.

C. Point of Conflict Detection

The edge server, on receiving the \( \text{aggregated}_p \) from the RSUs, extracts the \( \text{rsu}_i \) and the location of the RSU. Thereafter, it extracts the \( \text{encrypted}_p \) from the \( \text{aggregated}_p \) received. Furthermore, from the \( \text{encrypted}_p \), the symmetric key, \( K_{\text{v}i} \), for every vehicle is obtained by decrypting \( \tau \) with \( G_{\text{private}} \) using Algorithm 2 (lines 4-6). Finally, the obtained \( K_{\text{v}i} \) of a vehicle is further used to obtain \( \text{data}_{\text{packet}}' \) from \( \text{data}_{\text{packet}} \) (line 7). Thus, the response of a vehicle remains private from the RSU as well as from the nearby vehicles as \( G_{\text{private}} \) is only possessed by the edge server. From every \( \text{data}_{\text{packet}} \), the \( V_{\text{id}} \) is authenticated by the edge server by comparing \( ds(V_{\text{id}}) \) with the \( V_{\text{id}} \) and its associated \( \text{veh}_p \), which it receives from the centralized server (generated during the vehicle registration using EGDSS) as in Figure 4. The purpose of this step is to filter malicious vehicles that are performing any masquerading attack or identity theft. For all the authenticated vehicles, the velocity and the GPS location recorded by an individual \( V_{\text{id}} \) are obtained from the \( \text{data}_{\text{packet}} \). At first, the GPS location of a \( V_{\text{id}} \) is compared with the location of the RSU. If the GPS location is out of the transmission range of the RSU that records its data, this implies that the vehicle is manipulating its GPS location and it is filtered out as malicious. If not, the neighbors of a \( V_{\text{id}} \) are obtained by the edge server using Algorithm 2 (lines 7-11). Based on the neighbors extracted, the edge server constructs a DSG where every \( V_{\text{id}} \) forms a vertex of the DSG, and it has an undirected edge to its neighboring \( V_{\text{id}} \). This concept of undirected DSG is used to obtain the Point of Conflict (POC), defined as a situation where two neighboring vehicles, say \( V_i \) and \( V_j \), report a conflicting event like congestion and no-congestion, respectively within the same ROI (lines 15-20) and under the same RSU. The initial detection for POC is done using Algorithm 2, where the POC is searched within the neighbors (lines 13-22). If no POC is detected after the initial detection, there is still a chance for POC to exist among the vehicles within the same RSU (Algorithm 2 (lines 23-30)) if a malicious \( V_{\text{id}} \) intentionally chooses only malicious neighboring \( V_{\text{id}} \) to deceive the model from detecting any POC when analyzed based on its reported neighbors. Such a scenario has been further analyzed in Lemma 1.

We consider \( \text{Veh}_l \cap \text{Veh}_n \) to be a set of malicious \( V_{\text{id}} \), denoted by \( V_{\text{id}} \), whereas \( \text{Veh}_l \cap \text{Veh}_n \) is a set of non-malicious \( V_{\text{id}} \), denoted by \( V_{\text{id}} \). The cardinality of a set \( S \) is denoted by \( |S| \). \( \Gamma(V_{\text{id}}) \) represents a set of \( V_{\text{id}} \)’s present in the neighbor list of a \( V_{\text{id}} \) and \( \text{Veh}_l \cap \text{Veh}_n \cap \text{Veh}_l \cap \text{Veh}_n = 0 \), i.e., a \( V_{\text{id}} \) cannot be malicious and non-malicious at the same time.

Algorithm 2 Obtaining \( \text{data}_{\text{packet}} \) and Detecting POC

1: \( \text{key} = [0 - 9, A - Z, a - z] = [00 - 09, 10 - 35, 46 - 71] \)
2: \( \text{decid} = \text{decrypted encid} \)
3: \( \text{neighbor of } V_i \leftarrow \text{neighboring vehicle of } V_{\text{id}} \)
4: for every \( \tau \) received do
5: \( \text{Keyi} = \text{key}[\tau^G_{\text{private}} \mod (p \cdot q)] \)
6: end for
7: \( \text{data}_{\text{packet}} \leftarrow \text{decrypt data}_{\text{packet}} \) with \( \text{Keyi} \)
8: for every \( \text{encid} \) in \( \text{data}_{\text{packet}} \) do
9: \( \text{decid} \leftarrow \text{encid}^d_{\text{private}} \mod (p \cdot q) \)
10: \( \text{neighbor of } V_i \leftarrow \text{ascii-characters of} (\text{decid}) \)
11: end for
12: \( \text{POC detected} \leftarrow \text{false} \)
13: for every \( \text{data}_{\text{packet}} \) obtained from \( V_i \) do
14: for every \( V_j \) in \( \text{neighbor of } V_i \) do
15: if \( V_i, \text{event} \neq V_j, \text{event} \) then
16: \( \text{POC detected} \leftarrow \text{true} \)
17: \( \text{CV1} \leftarrow V_i \)
18: \( \text{CV2} \leftarrow V_j \)
19: go to Line 31
20: end if
21: end for
22: end for
23: for any 2 vehicles, \( V_k \) and \( V_m \) under same \( \text{rsuid} \) do
24: if \( V_i, \text{event} \neq V_m, \text{event} \) then
25: \( \text{POC detected} \leftarrow \text{true} \)
26: \( \text{CV1} \leftarrow V_k \)
27: \( \text{CV2} \leftarrow V_m \)
28: go to Line 31
29: end if
30: end for
31: return \( \text{POC detected}, \text{CV1}, \text{CV2} \)
Lemma 1. Given a $V_m^i$, it can have only $V_m^j$ s in its neighbor list if $\text{threshold} < |Veh\_list_m|$.

Proof. For $\text{threshold} < |Veh\_list_m|$, consider the value of $\text{threshold}$ to be $|Veh\_list_m|-1$, i.e., the maximum allowable $\text{threshold}$ under the constraint.

For a given $V_m^i$,

$$V_m^i \cup \Gamma(V_m^i) = Veh\_list_m,$$

the value of $\text{threshold} = |Veh\_list_m| - 1$, meaning that the neighbor list of a $V_m^i$ can include all other $V_m^j$ to avoid the detection of POC.

For $\text{threshold} > |Veh\_list_m|$, consider the value of $\text{threshold}$ to be $|Veh\_list_m| + 1$, i.e., the minimum allowable $\text{threshold}$ under the constraint.

This means a $V_m^i$ will have a $V_m^j$ in its neighbor list, under which the condition given below holds.

$V_m^i \cup \Gamma(V_m^i) \subset Veh\_list_m \cup Veh\_list_{nn}$

Thereafter, if no POC is detected, the server waits for the next $\text{threshold data packets}$ from the RSUs. If no POC is detected at all, it considers the similar event recorded by all the vehicles to be the event of the ROI. However, if a POC is detected, the edge performs initial scrutiny based on the information obtained from the vehicles in conflict as described below. During the initial scrutiny, the approximate velocity of a vehicle in a congested road is considered $vel_{congested}$, while the velocity in a non-congested road is considered $vel_{n-congested}$, with an allowable difference of $\epsilon$ mph to accommodate any minor variations in velocity.

1) If event recorded by a vehicle, say $V_i$, is congested, and its corresponding $vel_i$ is greater than $vel_{congested} + \epsilon$, then $V_i$ is considered malicious, and the $V_{id}$ in conflict with $V_i$ is considered non-malicious. Subsequently, the malicious vehicles are filtered from the network using Algorithm 5 (lines 11-14), and the decision is made based on the non-malicious vehicles.

2) If event recorded by a vehicle, say $V_i$, is non-congested, and its corresponding $vel_i$ is less than $vel_{n-congested} - \epsilon$, then $V_i$ is considered malicious, while the $V_{id}$ in conflict with $V_i$ is considered non-malicious. Subsequently, the malicious vehicles are filtered from the network using Algorithm 5 (lines 11-14), and the decision is made based on the non-malicious vehicles.

However, if no decision is made after initial scrutiny, the server generates a $\text{challenge_pkt}$ obtained using Algorithm 3 (line 23), consisting of $V_{id}$ in conflict and the RSUs under which the conflicting $V_{id}$ s are expected to appear after $time_{id}$ is calculated based on the $vel_{id}$ and $GPS_{id}$ recorded by the $V_{id}$ s. The purpose of the $\text{challenge_pkt}$ is to authenticate the $vel_{id}$ and $GPS_{id}$ recorded by the vehicle and to allow the $V_{id}$ s to prove its event recorded as it is assumed to travel with the same $vel_{id}$ for within the ROI.

$$\text{challenge_pkt} = \langle CV_1, \text{expected}_{rsu_1}, time_1, \text{CV}_2, \text{expected}_{rsu_2}, time_2 \rangle$$

Algorithm 3 Generating the $\text{challenge_packet}$

1: Initialize $t$
2: RSUList ← list of every rsu id
3: for every $CV_i$ do
4: calculated_distance ← $t \ast vel_{id}$
5: expected_location $i$ ← calculated_distance + $GPS_{id}$
6: for every $rsu_i \in RSU List$ do
7: if expected_location is within $rsu_i$ location then
8: if $CV_i$ event is "congested" then
9: $time_i = t$, $\text{expected}_{rsu_i} = rsu_i$
10: go to Line 23
11: else
12: $\text{expected}_{rsu_i} = rsu_i$
13: for every $rsu_i$ beyond $rsu_i$ do
14: $time_i = t$, $\text{expected}_{rsu_i} = \text{expected}_{rsu_i} + rsu_i$
15: end for
16: go to Line 23
17: end if
18: $t +=$
19: end if
20: end for
21: end for
22: $\text{challenge_packet} = CV_i + time_i + \text{expected}_{rsu_i}/s$
23: return $\text{challenge_packet}$

Based on the contents of the $\text{challenge_pkt}$s, the $\text{expected}_{rsu_1}$ should obtain a response from $CV_i$, i.e., the ID of one of the vehicles in conflict, after time, $time_1$, while the $\text{expected}_{rsu_2}$ should obtain a response from $CV_2$, i.e., the ID of the other vehicle in conflict, after time, $time_2$. To handle the case of overspeeding by a $CV_i$ recording "non-congested", every RSU along the direction of $CV_i$, obtained from its
velocity, dissipates the challenge_pkt (lines 13-15). This is done to ensure that even if a CV, passed by expected_rsus before timei, it still gets the challenge packet, as over speeding is only possible in case of a non-congested road scenario.

D. Challenge Packet Dissemination and Response

Upon receiving a challenge_pkt from the edge server, an RSU extracts its assigned CV, and timei. Thereafter, it generates a crypto_challenge packet and broadcasts it within its transmission range after timei. The crypto_challenge is generated based on the CV, assigned to it, and its purpose is to verify the presence of a vehicle within a specific region after time, seconds. This is leveraged in the proposed heuristic to filter the malicious vehicles within the network. Every vehicle at the ROI sends a crypto_response, defined as the unique response sent by V, in response to the crypto_challenge packet. The crypto_challenge packet is obtained by using bitwise manipulation (left shift operation) over the XOR cipher technique. The XOR cipher technique is computationally inexpensive and easy to implement. Furthermore, since every RSU dissipates the crypto_challenge packet generated using the XOR cipher technique exactly once in the proposed model, it is less susceptible to frequency analysis attacks and also man-in-the-middle attacks. The bitwise manipulation is to enhance the security of the crypto_challenge after the XOR operation.

crypto_challenge = CV, ⊕ (testing_word << left_num) left_num ∈ [1, length of testing_word - 1]
where testing_word is any arbitrary word chosen by a RSU having the same length as its assigned CV, and left_num refers to the number of left shift operations performed, which is chosen arbitrarily by the RSU.

Thereafter, upon receiving the crypto_challenge packet from the nearby RSU, every vehicle generates the crypto_response packet and broadcasts it. The purpose of the crypto_response packet is to validate the presence of CV, at a specific location, as calculated by the proposed heuristic. The crypto_response is a unique response to its associated crypto_challenge for a particular V, as generated by its corresponding expected_rsus.

crypto_response = V, ⊕ crypto_challenge

Algorithm 4 Analyzing crypto_response packets
1: Initialise response ← not received
2: for every crypto_response received by expected_rsus do
3:   if testing_word == (crypto_response >> left_num) then
4:     response = received
5:     break
6: end if
7: end for
8: vehicle_search = rsu_id + ";" + CV, + "" + response
9: return vehicle_search

The expected_rsus waits for additional σ seconds to receive the crypto_response packets from the V,gs. This is done to make reparation for a minor change in velCV, that may occur due to any trivial circumstance that does not affect the event at the ROI. However, it is assumed that velCV changes by a factor of atmost ε that still adheres to the decision recorded by CV. Thereafter, based on Algorithm 4 (lines 3-6), the RSU compares the crypto_response with the testing_word. The testing_word only matches with a specific CV (line 3). The associative and commutative nature of the XOR operation facilitates the effective analysis of the crypto_responses obtained.

Finally, the expected_rsus generates the vehicle_search packet using Algorithm 4 (line 8), which is sent to the edge server. The purpose of the vehicle_search packets is to report whether a CV, was present within the transmission range of expected_rsus within timei + σ.

vehicle_search =< rsu_id, CV, response >
where response can be received indicating that a CV, is present within the transmission range of expected_rsus within timei + σ, or not received which indicates that the vehicle was not present within timei + σ.

E. Decision Making by Edge Server

The edge server, on receiving the vehicle_search packets from expected_rsus, makes a decision about traffic conditions and filters malicious vehicles based on the heuristic depicted in Figure 5. According to the proposed heuristic:

1) If crypto_response has been received from one CV, and is not received from the conflicting CV, then the CV, from which the crypto_response has been received is considered non-malicious and the conflicting CV is considered malicious. Consequently, all the malicious vehicles with similar events recorded as the malicious CV, are filtered using Algorithm 5 (lines 11-14). The decision is made based on the decision of the non-malicious CV,.

2) If crypto_response has been received by both CV, then the edge server assumes that the CV, providing crypto_response with low vel, intentionally reduced its velocity to prove itself non-malicious. Under such a scenario, the decision made is "non-congested". Thereafter, every malicious vehicle is filtered using Algorithm 5 (lines 11-14).

However, if no POC is detected, then the traffic information about the ROI is the similar event recorded by the all the vehicles that have sent their responses to the edge server.

Fig. 5: Decision Tree for analysis
Algorithm 5 Filtering using DSG

1: CVi ← malicious vehicle detected, stack.push(CVi)
2: malicious_list ← malicious vehicle list
3: malicious_list.append(CVi)
4: non_malicious_list ← non malicious list
5: filtered[Vid] = false, filtered[CVi] = true
6: while stack not empty do
7:     CVi = stack.pop()
8:     for every Vid in DSG do
9:         if edge exists between Vid and CVi and filtered[Vid] == false then
10:             stack.push(Vid)
11:     if CVi.event == Vid.event then
12:         if CVi == malicious then
13:             Vid ← malicious
14:             malicious_list.append(Vid)
15:         else
16:             Vid ← non malicious
17:             non_malicious_list.append(Vid)
18:     end if
19:     else
20:         Vid ← non malicious
21:         non_malicious_list.append(Vid)
22:     end if
23: end if
24: end for
25: end while
26: return malicious_list, non_malicious_list

F. Response from Edge to Centralized Server

The traffic-related information from the ROI that has been identified by the edge servers is sent to the centralized server. The centralized server sends traffic-related information to the requesting vehicle in case of an ad hoc request, or the information is stored for traffic monitoring. The centralized server observes the traffic using the majority selection method and decides that the traffic scenario of multiple regions is the one reported by the majority of the edge servers. This is because the traffic condition in one region may have a cascading effect on the other regions. The centralized server in the proposed model is free from any bottleneck issues, which may have resulted on account of enormous requests about different ROIs. In the proposed model this is handled by different edge servers, and it also allows graceful degradation as the entire system will not falter if one edge server is faulty.

V. Experimental Results and Analysis

To validate the effectiveness of the proposed model, several experiments were conducted under both congested road conditions and non-congested condition under the influence of various attacks mentioned in the threat model using the simulation parameters described in Table I. The road network was simulated using SUMO [24], while the communication network was simulated using Omnet++ and VENTOS simulators. The centralized server (cloud) was simulated in a separate workstation, Windows 7 Enterprise 64 bit, Intel(R) Xeon(R) CPU E5-1620 v2 @ 3.70 GHz, and the edge server was simulated in a separate workstation, Windows 10, Intel Core i3-4005U CPU@ 1.7Ghz. In our simulation, we assumed roads that have a speed limit between 70 mph and 40 mph. In other words, the roads where vehicles travel below 35 mph are assumed to be congested whereas the roads where vehicles travel within the speed limit are non-congested. However, this can vary without any impact on the efficiency and utility of the proposed model. Based on the experiments performed under the parameters in Table I, it was observed that with the value of σ that were less than 10 seconds, the proposed model obtained the packets as desired. The responses of any of the CVi did not reach the nearby RSU within time with any values of σ lesser than 10 seconds. This is because on many occasions the channel remained busy transferring packets and because of this, the crypto_response had to wait in the pipeline as the simulator prevents any packet collision in the channel. The value of ϵ in the proposed model can be decided based on where it is deployed and can be adjusted without any change in the performance of the model. However, for the experimentation, we considered the value to be 5 mph. The proposed model has been compared against various other models using Detection_Accuracy as represented in Figure 6. Detection_Accuracy is a metric that is used to detect accurate road conditions under a varied percentage of malicious vehicles.

\[
\text{Detection}_\text{Accuracy} = \frac{\text{road condition}}{\text{% of malicious vehicles}}
\]

where Detection_Accuracy ∈ \{1,0.05\}

A value of 1 indicates that the accurate road condition and the malicious vehicles are detected while 0 indicates neither the road condition nor the malicious vehicles could be detected. A value of 0.5 indicates that the decision making is conditional, and it is either dependent on the prior reputation of vehicles (in the case of a reputation based system), the distribution of malicious vehicles (in the case of a peer authentication system) or has an equal percentage of malicious as well as non-malicious vehicles (in the case of a majority voting approach).

From Figure 6, it can be seen that the majority voting model and the peer authentication model have higher Detection_Accuracy when the majority of the vehicles within the ROI are non-malicious (greater than 50%). Under such cases, the decision of the non-malicious vehicles that are in the majority dominates the decision of the malicious vehicles. However, it becomes conditional when the number of malicious and non-malicious vehicles within the ROI are equal, as
the non-malicious vehicles do not form a majority, and hence no proper decision could be made. Thereafter, as the number of malicious vehicles increases beyond 50\%, it is seen that the Detection\_Accuracy decreases as the malicious vehicles form the majority under such scenarios. This influences the decision making process within the ROI. The Detection\_Accuracy of the Reputation\_Based model is always 0.5. This is because the decision is highly based on the reputation of the vehicles within the ROI. It is possible to have less malicious vehicles with a higher prior reputation (for instance, 5 malicious vehicles out of 100 vehicles with high rating) to dominate the majority of non-malicious vehicles with no prior reputation (for instance, 95 non-malicious vehicles out of 100 vehicles with no reputation). Thus, under such a system, the decision making model remains conditional. It is also to be noted that the number of vehicles within the ROI and $n_{rsu}$ represents the number of RSUs deployed within the ROI.

In the case of majority voting and reputation-based models, every vehicle within the ROI sends their response to the RSU that has a total of $n$ transmissions. Thereafter, the RSU sends threshold number of vehicular responses to the edge server at one time. Hence, all the packets are sent after $\frac{n}{\text{threshold}}$ times. Therefore, the total number of transmissions required is $n + \frac{n}{\text{threshold}}$.

In the peer authentication model, every vehicle at the ROI authenticates threshold number of vehicles. Therefore, the number of authentication transmissions are $n \times \text{threshold}$. Thereafter, the RSU sends all the packets to the edge server after $\frac{n\times\text{threshold}}{\text{threshold}}$ times, i.e., $n$ times. Thus, the total number of transmissions required in the peer authentication model is $(n\times\text{threshold}) + n$, which is equivalent to $n \times (1 + \text{threshold})$.

In our proposed model, every vehicle sends its $\text{enc}_id$ to its neighbor and thereafter, it sends the encrypted_packet to the RSU. Therefore, for every vehicle, it involves 2 transmissions. For $n$ vehicles, the total transmissions are $2 \times n$. The transmissions required by RSUs to send $n$ packets to the edge server is $\frac{n}{\text{threshold}}$. Thereafter, the initial scrutiny is performed. If the decision is made after initial scrutiny using the proposed model, we obtain the lower bound, Proposed Lower, on the transmissions, which is $(2 \times n) + \frac{n}{\text{threshold}}$. However, when the challenge\_response phase is executed, the challenge packet is sent to the $n_{rsu}$ RSUs by the edge server in one transmission, which is dissipated within the ROI. $n_{rsu}$ RSUs broadcast the crypto\_challenge packet and receive the crypto\_response packets from $n$ vehicles. Therefore, the total transmissions are $n + n_{rsu} + 1$. Finally, $n_{rsu}$ RSUs send their response to the edge server in $n_{rsu}$ transmissions. Under the challenge\_response phase, we obtain the upper bound on the transmissions of the proposed model, Proposed Upper, which is $(2 \times n) + \frac{n}{\text{threshold}} + 2 \times n_{rsu} + n + 1$. Based on the formulation, we find from Figure 8 that the total energy required for the transmission by the majority voting and reputation-based models are the least compared to the proposed model and the peer authentication model. This is because they involve no V2V communication. Therefore, with fewer number of broadcast, the transmission energy used is also less. However, our proposed model has fewer number of broadcast in both initial scrutiny (Proposed Lower) and when the challenge\_response is generated (Proposed Upper) than the peer authentication model because every vehicle has a fewer broadcast requirement, i.e., two broadcasts in Proposed Lower and three broadcasts in Proposed Upper. Hence, it requires less transmission energy compared to the peer authentication model, where every vehicle has to authenticate threshold number of neighboring vehicles. The time taken by our proposed model is highly dependant on the POC detection. It is noted from Figure 9 (plotted based on Table III) that the detection time for our model increases with the POC distance (the point where two vehicles conflict in event reporting as detected by the edge). This is because only when the POC is detected, the edge server begins the various steps in our
proposed model for detecting the road condition, and filtering out malicious vehicles. When the POC is detected early, i.e., within less POC distance, the edge proceeds with the initial scrutiny and thereafter, the challenge_response packet may be generated. However, with the increase in the POC distance, the edge server has to wait longer before executing the initial scrutiny phase. This can also be seen from Table III that the time taken is less dependant on the consecutive RSU distances compared to the POC distances during Proposed Lower. This is because even if the RSUs are close to each other, the edge server still has to detect a POC before the initial scrutiny phase is executed. The Proposed Upper with a consecutive RSU distance of 2500 and a POC distance of 10 in Table III takes less time (79 seconds) compared to the Proposed Lower with a consecutive RSU distance of 1000 and a POC distance of 20 (82.5 seconds). This is because in the latter scenario, the initial POC takes time to be detected by the edge. Therefore, a scenario having RSUs close to each other with a higher POC distance may take more time than a scenario with relatively distant consecutive RSUs but with less POC distance. Also, in the initial scrutiny phase, the time taken does not depend on the RSU distances (all the Lower values from Table III take the same time for detection for a given POC distance). This is because the vehicle responses is leveraged to make the decision after they are sent to the edge server. The RSU distance is impactul only when the challenge_response packet is generated (all the Upper values from Table III take different times for detection for a given POC distance). Through experiments, it was observed that the proposed model performs faster when the vehicles remain under some RSU throughout their travel. This is because if the vehicles are out of the transmission range of the RSU, it has to wait to arrive near the next RSU before sending their corresponding packets, which increases the latency. In the experiments, every vehicle remained under the transmission range of the RSU throughout their travel when they were placed 1000 metres apart, and it was observed that 1000 Upper as shown in Table III has the fastest performance when compared to 2500 Upper, 2000 Upper, and 1500 Upper, i.e., when the consecutive RSU distances were 2500 metres, 2000 metres, and 1500 metres apart, respectively. In Figure 9, Upper represents the total time taken for decision making when the challenge_response packet is generated, while Lower represents the time taken for decision making during the initial scrutiny phase. Table III

TABLE III: Time taken (in seconds) to detect traffic conditions with varying RSU and POC distances

| RSU Distance | POC Distance | 10 | 15 | 20 | 25 | 30 | 35 | 40 |
|--------------|--------------|----|----|----|----|----|----|----|
| 2500 Lower   | 21           | 53 | 84 | 104.0 | 190.3 | 256 | 326.8 |
| 2500 Upper   | 79           | 114 | 141 | 162.0 | 248.8 | 313 | 384.2 |
| 2000 Lower   | 23           | 54 | 84.33 | 102 | 193.85 | 254 | 327 |
| 2000 Upper   | 69           | 101 | 130.33 | 147 | 239.87 | 301 | 375 |
| 1500 Lower   | 22           | 55 | 82 | 101.2 | 192.5 | 257 | 327 |
| 1500 Upper   | 39           | 92 | 120 | 138.7 | 236.5 | 294.85 | 366 |
| 1000 Lower   | 24           | 55 | 82.5 | 101.2 | 194.2 | 257 | 326 |
| 1000 Upper   | 53           | 83 | 111.5 | 132.2 | 222.2 | 286 | 354 |

Fig. 9: Broadcast comparison of 100 vehicles with varying thresholds

also reveals the limitation of the proposed model. It can be seen that the time taken to detect the traffic condition is not in real time. This is because the amount of time lapsed in sending the data_packets and detecting the traffic condition is not real time. Thus, this restricts the application of the model in scenarios that do not require real time decisions. This is justified by the fact that the ROI is usually very large, and the main purpose of the proposed model is mainly to
allow the vehicles in one ROI to know the traffic scenario of another ROI that it wants to enter. However, the proposed model cannot be applied in autonomous vehicles that require decisions in negligible time, such as turning the steering wheel or braking on the road. Based on the values in Table III, the detection probability (defined as the ratio of the number of observations where the road condition is detected within a time limit to the total number of observations) is represented in Figure 10. It is to be noted that as the proposed model drifts away from real time, detection probability increases. This is because with more time, conflicting vehicles at a higher POC distance are also be considered which increase the accuracy in the proposed model. Hence, the probability of finding a POC and detecting more scenarios increases as more POC distances are covered, and eventually, it leads to an increase in the detection probability at the cost of time. For example, in Table III, the number of observations with the time limit below 50 seconds is 4 (21,23,22,24). This means that out of 56 observations in Table III, only 4 observations (50% of the observations with POC distance 10) detect within 50 seconds, thereby covering approximately 7% of the total observations that detect the road condition with a detection probability of 0.07. However, it is further noted from Table III that the number of observations that detect below 100 seconds is 16, which covers every observation with the POC distance of 10, 80% of the observations with the POC distance of 15, and approximately 50% of the observations with the POC distance of 20. This covers almost 28.6% of the total observations in Table III (with a detection probability of 0.28) as compared to 7%, when the time limit was below 50 seconds. Thus, with more time, a larger POC distance is considered in the proposed model, which results in an increase in the detection probability.

\[ \text{detection probability} = \frac{\text{total detections under certain time}}{\text{total possible detections}} \]

VI. SECURITY ANALYSIS

A. Message and GPS information spoofing attack

In our proposed model, every vehicle records an event and generates a data packet. In order to disrupt the decision making process, it can report an inaccurate event, i.e., report a congestion when the road is non-congested. For the attack to be successful, every vehicle at the ROI must spoof the event, which would incapacitate the edge server to detect any POC. However, this contradicts our assumption that at least one vehicle must be non-malicious. Furthermore, an attacker may spoof the GPS information, i.e., sent manipulated vel\text{ld} and GPS\text{ld} to the edge server. However, in the proposed heuristic, the edge server leverages vel\text{ld} and GPS\text{ld} to generate the challenge\_packet to filter malicious vehicles within the ROI. Thus, our proposed model is secure against message and GPS information spoofing attack.

B. Masquerading, Collusion and Sybil attack

In the proposed model, every vehicle digitally signs its V\text{id} using veh\_private, generated using EGDSS to produce ds(V\text{id}). An attacker can forge the signature if it can compute veh\_private, of the attacked V\text{id} or by performing hash collision attack. However, SHA-3 [25] is secure against a collision attack, preimage [26] and second preimage attack[27] with the security strength varying from 112-256 bits for collision and 224-512 bits for preimage and second preimage attack. An attacker may also perform collusion or a sybil attack to disrupt the decision making process. However, for the attack to be successful, every V\text{id}s within the ROI must be malicious. This contradicts our assumption of having one non-malicious vehicle within the ROI. Thus, the proposed model is secured against masquerading, collusion and sybil attacks.

C. Conditional Privacy Preservation

In our proposed model, every vehicle generates enc\_id by encrypting its V\text{id} with G\text{public}. If an attacker attempts to track the V\text{id} of a vehicle, it has to compute G\text{private} using the associated value of prime numbers, p and q, possessed only by the edge server. In order to breach the privacy of the data packet of a vehicle, it has to obtain the Key\text{id}, generated using AES 128, used by a V\text{id} to encrypt the data packet. Even under such a scenario, the attacker has to compute G\text{private} as Key\text{id} is encrypted using G\text{public}. Thus, the proposed model guarantees anonymity and unlinkability of a vehicle. However, the edge server filters malicious vehicles within the ROI using the DSG. Thereafter, it is stored in the centralized server for future reference. Thus, the proposed model also guarantees conditional privacy and only reveals the identity of the malicious vehicles when it detects a conflict.

D. Message Integrity Attack

In the proposed model, every vehicle sends encrypted_data_packet =< τ, data_packet > to an edge server. If an attacker wants to violate the integrity of a data_packet generated by a V\text{id}, he/she must acquire G\text{private} stored only at the edge server. This means that an attacker has to compute p and q used by the edge server to generate G\text{private}. However, this violates the discrete logarithm problem [28]. An attacker may also try to compute Key\text{id} generated using AES 128 to obtain data_packet. However, a fastest supercomputer operating at 10.51 petaflops performing around 1000 checks per second approximately takes around 1 billion years to brute-force the key [29] while the biclique attack [30] requires a computational complexity of \(2^{126.1}\).
which is highly unlikely to break in real-time. Thus, our proposed model is resilient against message integrity attack.

E. DoS attack

Let us assume that the number of vehicles within the ROI is $V_{\text{num}}$. The various combinations of malicious vehicles not sending the packet to the edge server, defined as C(DoS), is given by:

$$C(\text{DoS}) = \sum_{i=1}^{V_{\text{num}}} \binom{V_{\text{num}}}{i}$$

C(DoS) represents the different number, ranging from 0 to $V_{\text{num}}$, of vehicles that can refrain from sending the information to the edge server via the RSU. However, DoS is possible only when every vehicle within the ROI drops their data packets. Thus, the probability of the DoS attack being successful, defined as $P(\text{DoS})$, contradicts our assumption that every non-malicious vehicle sends their packets, at least one packet will be received by the edge server, and also that one non-malicious vehicle should be present within the ROI. Thus, the proposed model prevents DoS attacks.

$$P(\text{DoS}) = \frac{1}{C(\text{DoS})}$$

VII. CONCLUSION

In this paper, we proposed a privacy preserving secure edge cloud-assisted traffic monitoring system for VANETs that provides accurate traffic-related information within the ROI. The proposed model is resilient against privacy attacks and unauthorized tracking and is secured against collusion, masquerading, ballot stuffing, and bad mouthing attacks. We introduced DSG and the challenge-response strategy to filter malicious responses within the ROI to determine accurate traffic-related information with fewer number of broadcasts per vehicle compared to the peer authentication model. Even though the number of broadcasts per vehicle required for the proposed model is higher than the majority voting model and the reputation based model, the proposed model has a higher detection accuracy when the number of malicious vehicles forms the majority within the ROI. This means that unlike the majority voting and the reputation-based model, the proposed model filters malicious vehicles and accurately detects the traffic condition of the ROI under the influence of at least one non-malicious vehicle.

REFERENCES

[1] Zhe Yang, Kan Yang, Lei Lei, Kan Zheng, and Victor CM Leung. Blockchain-based decentralized trust management in vehicular networks. IEEE Internet of Things Journal, 2018.
[2] Jun Shao, Xiaodong Lin, Rongxing Lu, and Cong Zuo. A threshold anonymous authentication protocol for vanets. IEEE Transactions on vehicular technology, 65(3):1711–1720, 2015.
[3] Philippe Golle, Dan Greene, and Jessica Staddon. Detecting and correcting malicious data in vanets. In Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks, pages 29–37. ACM, 2004.
[4] Qin Li, Amizah Malip, Keith M Martin, Siaw-Lynn Ng, and Jie Zhang. A reputation-based announcement scheme for vanets. IEEE Transactions on Vehicular Technology, 61(9):4095–4108, 2012.
[5] Andy Greenberg. Hackers remotely kill a jeep on the highway with me in it. Wired, 7:21, 2015.
[6] William Whyte, André Weimerskirch, Virendra Kumar, and Thorsten Hahn. A security credential management system for v2v communications. In 2013 IEEE Vehicular Networking Conference, pages 1–8. IEEE, 2013.
[7] Maram Bani Younes and Azzeddine Boukerche. Scool: A secure traffic congestion control protocol for vanets. In 2015 IEEE Wireless Communications and Networking Conference (WCNC), pages 1960–1965. IEEE, 2015.
[8] Jezabel Molina-Gil, Pino Caballero-Gil, and Cándido Caballero-Gil. Congestion and probabilistic verification for data authentication in vanets. Information Sciences, 262:172–189, 2014.

[9] Shereen A Taie and Sanaa Taha. A novel secured traffic monitoring system for vanet. In 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), pages 176–182. IEEE, 2017.
[10] Ahren Studer, Fan Bai, Bhargav Bellur, and Adrian Perrig. Flexible, extensible, and efficient vanet authentication. Journal of Communications and Networks, 11(6):574–588, 2009.
[11] Shrikkant Tangade, Sunilkumar S Manvi, and Pascal Lorenz. Decentralized and scalable privacy-preserving authentication scheme in vanets. IEEE Transactions on Vehicular Technology, 67(9):8647–8655, 2018.
[12] Joos Daemen and Vincent Rijmen. The block cipher rijndael. In International Conference on Smart Card Research and Advanced Applications, pages 277–284. Springer, 1998.
[13] Diaa Salama Abd Elminna, Hatem Mohamed Abdual-Kader, and Mohiy Mohamed Hadhoud. Evaluating the performance of symmetric encryption algorithms. IJ Network Security, 10(3):216–222, 2010.
[14] Karita Schmidt-Samoa. A new rabin-type trapdoor permutation equivalent to factoring. Electronic Notes in Theoretical Computer Science, 157(3):79–94, 2006.
[15] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 21(2):120–126, 1978.
[16] Michael O Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical report, Massachusetts Inst of Tech Cambridge Lab for Computer Science, 1979.
[17] Neal Koblitz. Elliptic curve cryptosystems. Mathematics of computation, 48(177):203–209, 1987.
[18] Krists Magons. Applications and benefits of elliptic curve cryptography. In S0FSEM (Student Research Forum Papers/Posters), pages 32–42, 2016.
[19] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE transactions on information theory, 31(4):469–472, 1985.
[20] David W Kravitz. Digital signature algorithm, July 27 1993. US Patent 5,231,668.
[21] Hyo Jin Jo, In Seok Kim, and Dong Hoon Lee. Reliable cooperative authentication for vehicular networks. IEEE Transactions on Intelligent Transportation Systems, 19(4):1065–1079, 2017.
[22] Nuno Santos, Krishna P Gummadi, and Rodrigo Rodrigues. Towards trusted cloud computing. HotCloud, 9(3):3, 2009.
[23] Zhen-Hui Zha and Elizabeth G Jones. Transmission range evaluations of connected vehicles at highway-rail grade crossings. Designs, 1(1):2, 2017.
[24] Michael Behrisch, Laura Bieler, Jakob Erdmann, and Daniel Krajzewicz. Sumo—simulation of urban mobility: an overview. In Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation. ThinkMind, 2011.
[25] Morris J Dworkin. Sha-3 standard: Permutation-based hash and extendable-output functions. Technical report, 2015.
[26] Kazumaro Aoki and Yu Sasaki. Preimage attacks on one-block md4, 63-step md5 and more. In International workshop on selected areas in cryptography, pages 103–119. Springer, 2008.
[27] Hongbo Yu, Gaoli Wang, Guoyan Zhang, and Xiaoyun Wang. The second-preimage attack on md4. In International Conference on Cryptology and Network Security, pages 1–12. Springer, 2005.
[28] Kevin S McCurley. The discrete logarithm problem. In AMS Proc. Symp. Appl. Math, volume 42, pages 49–74, 1990.
[29] Mohit A Arora. How secure is aes against brute force attacks. EE Times, 5(7), 2012.
[30] Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. Biluclidean cryptanalysis of the full aes. In International Conference on the Theory and Application of Cryptology and Information Security, pages 344–371. Springer, 2011.