SACRIFICE: A Secure Road Condition Monitoring Scheme over Fog-based VANETs

Nishttha Sharma
Indian Institute of Engineering Science and Technology
Howrah, India
Email: nishttha.sharma@gmail.com

Jayasree Sengupta
Indian Institute of Engineering Science and Technology
Howrah, India
Email: jayasree202@gmail.com

Sipra Das Bit
Indian Institute of Engineering Science and Technology
Howrah, India
Email: sdasbit@yahoo.co.in

Abstract—With the rapid growth of Vehicular Ad-Hoc Networks (VANETs), huge amounts of road condition data are constantly being generated and sent to the cloud for processing. However, this introduces a significant load on the network bandwidth causing delay in the network and for a time-critical application like VANET such delay may have severe impact on real-time traffic management. This delay maybe reduced by offloading some computational tasks to devices as close as possible to the vehicles. Further, security and privacy of vehicles is another important concern in such applications. Thus, this paper proposes a secure road condition monitoring scheme for fog-based vehicular networks. It considers Road-Side Units (RSUs) with computational capabilities, which act as intermediate fog nodes in between vehicles and cloud to reduce delay in decision making and thereby increases scalability. Apart from fulfilling basic security features, our scheme also supports advanced features like unlinkability and untraceability. A detailed security analysis proves that the proposed scheme can handle both internal and external adversaries. We also justify the efficiency of our scheme through theoretical overhead analysis. Finally, the scheme is simulated using an integrated SUMO and NS-3 platform to show its feasibility for practical implementations while not compromising on the network performance. The results show an average 80.96% and 37.5% improvement in execution time and end-to-end delay respectively while maintaining at par results for packet delivery ratio over a state-of-the-art scheme.

Index Terms—Fog Computing, Authentication, Unlinkability, Untraceability.

I. INTRODUCTION

With the growth of the modern society along with the rise of smart cities, the number of connected devices and vehicles have increased exponentially. This has prompted the development of Vehicular Ad-Hoc Networks (VANETs) which not only facilitates traffic management but also makes driving safer by exploiting the inter-vehicular communication feature [1]. As a result of this, huge amounts of data are being generated every second which needs to be processed in a time-sensitive manner. This has inspired the introduction of fog computing [2] in VANETs to reduce transmission delay, network bandwidth and improve overall traffic management. By extending the cloud closer to the vehicles in the network, fog has been able to reduce delay in decision making thereby improving real-time road condition monitoring, real-time driving assistance etc [3]. However, introduction of fog in VANETs may incur additional security threats apart from the inherent ones, including misuse or leakage of sensitive data which needs to be handled with utmost care.

The existing literature have focused on introducing fog as a middleware in VANETs to utilise its benefits. The works [4, 5] have introduced a layer of fog devices between vehicles and cloud to provide improved network throughput, reduced latency and increased scalability. However, even after introducing fog several security and privacy issues, data quality and scheduling related challenges still exists. To address these concerns, the existing works focus on managing various security aspects in VANETs. The works [6–8] propose different methods for mutual authentication by using securely agreed session keys and/or certificateless aggregate signcryption techniques to support privacy protection. But, it has high computation-communication overheads. On the contrary, the works [9, 10] have proposed an identity-based mutual authentication protocol with only hash functions and Ex-OR operations which reduces overheads significantly. Recently, the works [11, 12] have proposed an authentication protocol for VANETs with password and group key agreement. However, most of these works only concentrate on securing the vehicle to RSU communication. Finally, the work [13] has proposed a secure road condition monitoring scheme with authorized reporting, privacy-preserving monitoring, and source authentication. However, their scheme is cloud-based which causes increased latency for a system like VANET. It also does not take into consideration the privacy of the vehicle user and a few other security features like resistance to various attacks such as man-in-the-middle, replay etc.

From the above discussion it is clear that the state-of-the-art works rely largely on a cloud-based platform which results in a higher end-to-end delay. Further, despite the fact that these works focus on certain security features like user anonymity, mutual authentication, non-repudiation etc. various other important security features still remain less investigated in such VANET environments. This motivates us to design an improved architecture for VANETs and propose a Secure roAd Condition monitoRIng scheme over Fog-based vehICular nEtworks (SACRIFICE) keeping in mind the application’s time-sensitive nature. The SACRIFICE also targets to incorporate security features (e.g. unlinkability to prevent attackers from tracking the movement of vehicles) in
addition to basic security while running the scheme. Therefore, the contributions put forth by our work are as follows:

- We propose SACRIFICE having the following features:
  - reduce delay in decision making by considering Fog-based VANET while maintaining basic security features like mutual authentication, user anonymity etc.
  - introduce additional security features like non-repudiation, unlinkability and untraceability.
- A detailed security analysis proves that SACRIFICE can handle both internal and external adversaries.
- We validate SACRIFICE both theoretically and experimentally.
  - Establish SACRIFICE to be lightweight as well as having low overheads compared to state-of-the-art works.
  - Simulation results in an integrated real-time platform using SUMO and NS-3 establish the practicality of the scheme.

The rest of the paper is structured as follows. Section II discusses the system model. SACRIFICE is presented in Section III. Section IV briefly explains the security analysis of the scheme. Section V highlights the performance of the scheme. Finally, Section VI concludes the work.

II. SYSTEM MODEL

This section illustrates the system model in detail where the fog-based VANET architecture inspired from \[3, 13\] is used as the backbone of our work. Followed by that, we explain the security guarantees and the adversarial model.

A. Architecture

The architecture shown in Fig. 1 comprises four layers and has inculcated the advantageous features of \[3, 13\]. Each of these layers has entities like vehicles, RSUs etc. and can communicate with its immediate upper and lower layers. The activities of these four layers are described below:

Vehicular Network Layer: It consists of vehicles equipped with various sensors (camera, temperature, etc.) \[14\] and On-Board Units (OBUs) which has communication capabilities. The vehicles are responsible for gathering the sensory data and sending this information along with the location and time to the fog layer. In turn the vehicles receive information from fog nodes in case of a service request.

Fog Layer: It consists of Road-Side Units (RSUs) enabled with computing capabilities. These are installed at important junctions of the roads maintaining specific distance from each other depending upon the communication range to provide maximum coverage while guaranteeing persistent links with the cloud. RSUs use Dedicated Short-Range Communication to interact with the vehicles. These nodes are more robust and are responsible for minimizing the delay in decision making by extending the cloud closer to the vehicular network layer \[2\]. An RSU receives sensory data from various vehicles within its range, processes it and sends it to the cloud for further action.

Cloud Layer: It typically consists of different kind of storage or application servers which are responsible for communicating with the fog layer in order to receive data about the entire network. The cloud may process such data and forward certain reports to the Application Authority, if required.

Application Layer: It consists of the Trusted and the Application Authorities. The Trusted Authority is responsible for device registration (e.g. vehicles) and key distribution or application servers which are responsible for communicating with the fog layer in order to receive data about the entire network.

B. Security Guarantees and Adversarial Model

The security features, adversarial model and assumptions considered in SACRIFICE are discussed here.

Security of the Scheme

The following are the security requirements of the scheme which are taken care of:

Mutual Authentication \[3, 7\]: The validity of all participants (i.e. vehicles, RSUs) needs to be guaranteed. This requires that none of the malicious participants should be able to impersonate some other valid participant without being detected. Thus, vehicles and RSUs should authenticate each other to prevent forgery of tokens exchanged between them.

User Anonymity and Untraceability \[3, 7\]: It requires protecting the vehicle users’ privacy during data transmission to hide its real identity and behavioural patterns from the network. This means that an attacker intercepting the messages cannot extract a user’s real identity or track its behaviour (e.g. route, driving patterns).

Non-Repudiation \[3\]: The scheme guarantees that vehicles shouldn’t be able to deny its involvement in case of any dispute (e.g. sending corrupted data). It implies that in case of denial from the vehicles, the RSUs will be able to prove the role of the vehicles to any third party.

Unlinkability \[10\]: This feature prevents an adversary from determining whether two messages \((m_1, m_2)\) have originated from the same vehicle or not. Thus, an adversary will be unable to link messages generated by the same vehicle and thereby fail to distinguish between vehicles.
Resistance to common attacks [7]: To ensure security of the scheme, it is important to prevent attacks like man-in-the-middle and replay attacks [15]. For example, an attacker neither can pretend to be a legitimate user to cheat other participants nor can it launch an attack on the scheme to continue sending old messages to overload the network.

Adversarial Model

Entity: An entity can either be honest, semi-honest or malicious. Semi-honest entities do not deviate from the protocol specifications but may intend to obtain intermediate results/information from the nearby entities. On the contrary, malicious entities may deviate from the protocol arbitrarily.

Adversary: An adversary is a polynomial-time algorithm that can compromise any party at any point of time, subject to some upper bound [16]. Adversaries can be broadly categorized into two types: internal and external. External adversaries do not possess authentic keying material and hence they cannot participate as valid nodes [17]. They can only eavesdrop on radio transmissions and try to access information from the data transmitted through the channels. On the contrary, internal adversaries possess authentic keying material and have more effective and powerful resources in terms of energy and communication capabilities and are more vulnerable than external adversaries [17]. When an internal adversary captures a device in the network, it means the adversary has gained control over the tokens (not stored in a tamper-proof box) in the device. It also gains control over the messages sent/received by the device. We assume an adversary can neither interfere with the message exchanges between honest parties nor can it break cryptographic primitives like hash functions, except with a negligible probability. In this work, we consider both external and internal adversaries, however they have bounded computational and storage capabilities.

Assumptions: The following assumptions are made while setting up the proposed scheme:

- There is a secure channel between the Application Authority and the Trusted Authority as well as between the Trusted Authority and a device that is being registered.
- The vehicles can be malicious, RSUs are semi honest and Cloud is an untrusted entity.
- The Application Authority and the Trusted Authority are honest and trusted entities.

III. PROPOSED SCHEME

A detailed overview of our proposed Secure roAdd Condition monitoRING scheme over Fog-based vehICular networks (SACRIFICE) along with its algorithmic constructs is discussed in this section. The scheme generates a road condition report through a distributed process running in the participating vehicles and the roadside units of the network with the RSUs performing the intensive computations.

Working Principle: Whenever a vehicle \( U_i \) enters the scope of an RSU \( R_j \), both the devices have to mutually authenticate each other. After successful authentication, \( U_i \) sends an initial road condition report to \( R_j \) from which it generates the final road condition report. This final report is sent to the cloud for storage and further processing. The report can be extracted by the Application Authority to make important decisions in case of an emergency. To ensure honest behaviour of the participants, a hash-based lightweight mutual authentication algorithm has been implemented. Our proposed scheme consists of five different phases which are outlined below.

![SACRIFICE](image)

**Phase 1: System Setup**

This phase is performed during the establishment of the network. In this phase, each of the parties involved in the communication converge on the security parameters \( prms = (q, G, P, P_{pub}, h_1, h_2, h_3, h_4, h_5, h_6, h_7, \tau) \).

**Phase 2: Device Registration**

All the devices, i.e. both vehicles and RSUs are registered prior to entering the network.

**Phase 3: Mutual Authentication**

The mutual authentication between the vehicles and the RSUs is inspired from [7] [10], however we have modified their schemes to eliminate interactions with the Trusted Authority.
(TA) during this phase. This reduces the trust dependence on any third party entity during the execution of SACRIFICE.

When a vehicle $U_i$ enters the scope of an RSU $R_j$, the $U_i$ sends out the required tokens as authentication request to the $R_j$. On receiving the authentication request, RSU checks its validity. For a successful validation, $R_j$ generates the required tokens as authentication response and sends it to $U_i$. In turn, $U_i$ checks the validity of the authentication response. On successful validation it is said that both the vehicle and the RSU have mutually authenticated each other and can proceed further. On failure, of any of the above steps, the protocol is terminated.

### Phase 4: Report Generation

We adopt report generation and processing method from [13]. But unlike [13], here we divide the report generation task between the RSUs and vehicles instead of getting it performed by the vehicles alone. This report generation is broken down into two sub-tasks: \textit{initial report} and \textit{final report} generations respectively. This substantially reduces the delay of the entire scheme and improvises faster decision making by the Application Authority (AA).

In this phase a vehicle may generate an initial road condition report and send it to its nearest RSU. The RSU in turn processes the report to extract necessary information. It then generates a final road condition report which is sent to the Cloud Server (CS).

### Phase 5: Report Processing

When the Cloud Server receives a report from an RSU, it first checks the validity of the report and stores it in an appropriate equivalence class, if the report is valid. Here, equivalence class refers to a class consisting of a set of tuples which report the same road condition for the same location within a reasonable time period [13]. If the targeted
equivalence class reaches a predefined threshold $\tau$, a report from that particular class is sent to the Application Authority (AA) for further processing. The AA in turn tests the validity of the report and based on it decides whether the report is to be accepted or not. If accepted, it extracts the road condition information from the report.

Fig. 2 gives a first level view of the entire scheme whereas the tasks performed by each of the entities mentioned in this figure are elaborated through the respective procedures provided next. Here, the functions executed by Trusted Authority (TA), Vehicle $U_i$, RSU $R_j$, Cloud Server (CS) and Application Authority (AA) are explained in detail in Procedures 1, 2, 3 and 4 respectively. Referring to the figure and the procedures, at any instance of time many such $(U_i, R_j)$ pair can communicate. Hence, our scheme fits well to a multiple vehicle-RSU setting as well.

IV. SECURITY ANALYSIS

This section analyzes the security of SACRIFICE.

A. Mutual Authentication

In SACRIFICE, when a malicious vehicle $U_i$ attempts to mutually authenticate itself with a legitimate RSU $R_j$, the following computations are performed:

Let $U_i$ select a random $r_i \in Z_q^*$, a forged identity $ID_{U_i}$, and derive the current timestamp $t_{U_i}$. It then calculates the following:

$$X_1 = r_i \cdot P$$
$$X_2 = r_i \cdot P_{pub}$$
$$X_3 = h_2(X_1, X_2, t_{U_i}) \oplus ID_{U_i}$$

However, to calculate $C_i$ it must have the value of $P_{U_j}$, which is only sent to vehicles during registration. Without the value of $msk$, $U_i$ cannot calculate $P_{U_j}$. Let us assume that $U_i$ selects a random value $P'_{U_j}$, and computes:

$$C_i = h_3(ID_{U_i}, ID_{R_j}, P'_{U_j}, X_1, X_2, t_{U_i})$$

and sends $M_1 = \{X_1, X_3, C_i, t_{U_i}\}$ as the authentication request to $R_j$. On receiving $M_1$, $R_j$ calculates:

$$ID_{U_i} = X_3 \oplus h_2(X_1, X_2, msk, t_{U_i})$$
$$ID'_{U_i} = h_3(ID_{U_i}, ID_{R_j}, h_1(msk', ID_{U_i}), X_1, X_3, msk, t_{U_i})$$

However, on checking $R_j$ finds that $C_i'$ is not equal to the received $C_i$ in $M_1$ (since $P_{U_j} \neq h_1(msk, ID_{U_i})$). Thus, $R_j$ rejects the authentication request. Hence, it is proved that SACRIFICE doesn’t allow any malicious vehicle to operate in the network.

B. User Anonymity and Untraceability

1) User Anonymity: An adversary $A$ (internal or external) may try to obtain the real identity of vehicle $U_i$, where the following two cases may arise:

Case 1: Let us assume, an external adversary $A$ intercepts message $M_1$ and obtains the value of $h_{U_i}$. It also obtains the value of $X_3 = h_2(X_1, X_2, t_{U_i}) \oplus ID_{U_i}$, where $X_1 = r_i \cdot P$ and $X_2 = r_i \cdot P_{pub}$. To extract the real identity $ID_{U_i}$ of $U_i$ from $X_3$, $A$ needs the value of $X_2$ which can be calculated either using $r_i$ or $msk$, both of which are unknown to $A$ and also cannot be evaluated by it in polynomial time. Similarly, $A$ also cannot extract $ID_{U_i}$ from $C_i = h_3(ID_{U_i}, ID_{R_j}, P_{U_i}, X_1, X_2, t_{U_i})$ since it requires $r_i$ for the same, which is unknown to it.

Case 2: Let us now consider that $A$ is an internal adversary and it takes control over the RSU $R_j$. It is already known that $A$ cannot acquire the keys $msk$ and $P_{R_j}$. It can however gain control over the messages sent/received by $R_j$. Thus, when $U_i$ sends the authentication request message $M_1$, $A$ intercepts $M_1$ and computes $ID_{U_i}$ as below:

$$ID_{U_i} = X_3 \oplus h_2(X_1, X_2, msk, t_{U_i})$$

Since, $msk$ cannot be accessed by $A$, the value of $ID_{U_i}$ cannot be calculated as well. Thus, $A$ assumes a random value $msk'$ in place of $msk$ and calculates

$$ID'_{U_i} = X_3 \oplus h_2(X_1, X_2, msk', t_{U_i})$$

Since $A$ controls $R_j$, therefore $R_j$ sends $M_2 = \{Y_1, C'_j, ID_{R_j}, t_{R_j}\}$ to $U_i$ where,

$$C'_j = h_3(ID_{U_i}, h_1(msk', ID_{U_i}), ID_{R_j}, Y_1, Y_2, t_{R_j})$$

On receiving $M_2$ from $R_j$, $U_i$ calculates

$$C_j = h_3(ID_{U_i}, P_{U_i}, ID_{R_j}, Y_1, Y_2, t_{R_j})$$

However the calculated $C_j$ value mismatches the $C'_j$ extracted from received $M_2$ (since $ID'_{U_i} \neq ID_{U_i}$ and $P_{U_i} \neq h_1(msk', ID'_{U_i})$). Therefore, $U_i$ terminates the authentication process thereby preserving user anonymity.

2) Untraceability: An adversary $A$ attempting to know the behaviour of a participant $R_j$ is discussed here. The messages sent from $R_j$ are $M_2$ and $M_4$. Here, $M_2$ is dependent on the random values $r_i$ and $r_j$ as well as timestamps $t_{U_i}$ and $t_{R_j}$. $M_4$ is dependent on the session key $snky$ and also on timestamps $t_{U_i}$ and $t_{R_j}$. The value of these random variables and timestamps are different for each instance of the respective messages. Therefore, every time a message is sent, its contents change due to their dependency on these dynamic variables making it difficult to understand the behaviour of $R_j$. For example, two vehicles entering the range of $R_j$ at different timestamps attempt mutual authentication with it. $R_j$ then chooses two random $r_j$ values, for authentication with vehicles $U_1$ and $U_2$. Thus, the contents of the message (say $M_2$) will be different for $U_1$ and $U_2$ not only because they are sent out at different timestamps but also because of the presence of different random values in them.

C. Non-Repudiation

During mutual authentication, $R_j$ can calculate the real identity of vehicle $U_i$ from message $M_1$ as:

$$ID_{U_i} = X_3 \oplus h_2(X_1, X_2, msk, t_{U_i})$$

and also, from $M_3$ as:

$$ID_{U_i} = Q_1 \oplus h_5(t_{U_i}, snky, ID_{R_j})$$

This is useful to prove the involvement of a vehicle, if the vehicle denies participation in case of any malicious behavior.

D. Unlinkability

When a vehicle $U_i$ first enters the range of an RSU $R_j$ and after sometime, in the range of another RSU $R_k$, an authentication request is sent by $U_i$ to the respective RSUs. $U_i$ sends $M_1$ and $M'_1$ to $R_j$ and $R_k$ respectively. We consider both these authentication requests have been fulfilled successfully.
According to SACRIFICE, \( U_i \) will choose a new random value for \( r_i \) before sending out a new authentication request. Therefore, the tokens in \( M_i \) dependent on these values will also change with every authentication request.

For example, if \( M_i = \{X_1, X_3, C_{X_2}, t_{U_i}\} \) then \( M'_i = \{X'_1, X'_3, C'_{X_2}, t'_{U_i}\} \), i.e., all the tokens of the authentication request message will change. So, when an attacker \( A \) intercepting the messages \( M_i \) and \( M'_i \) from a public channel, cannot link the messages to the same vehicle as the contents of both the messages are different.

E. Resistance to common attacks

In this section, we discuss resistance against two of the most common attacks: (a) Man-in-the-Middle (b) Replay Attacks.

1) Man-in-the-Middle Attacks: Here, we assume that an adversary \( A \) (internal or external) already knows the values of \( ID_{U_i} \) and \( ID_{R_j} \) and wishes to forge a message. The following two cases may arise:

**Case 1:** When an external adversary \( A \) tries to forge message \( M_2 \), it chooses some random value \( r'_i \in Z_q^* \) and calculates \( Y_1 = r'_i \cdot P \) and \( Y_2 = r'_j \cdot X_1 \cdot P_{pub} \). However, in order to calculate \( C_j \), it requires the value of the master key \( msk \) which is unknown to \( A \). Moreover, \( C_j \) cannot be calculated in polynomial time. Hence it assumes a random value \( msk' \) in place of \( msk \) and computes \( C_j' \) as follows:

\[
C_j' = h_A(ID_{U_i}, h_1(msk', ID_{U_i}), ID_{R_j}, Y_1, Y_2, t_{R_j})
\]

On receiving \( M_2 \), \( U_i \) computes \( C_j \) and finds out that the calculated \( C_j \) is not equal to \( C_j' \) received in \( M_2 \) (since \( P_{U_i} \neq h_1(msk', ID_{U_i}) \)) and thereby terminates the authentication process. Thus, it is proved that an external adversary \( A \) cannot successfully forge message \( M_2 \).

**Case 2:** We consider an internal adversary \( A \) has gained control over RSU \( R_j \) and tries to forge message \( M_2 \). As per SACRIFICE, we already know that \( A \) cannot acquire the keys \( msk \) and \( P_{R_j} \) but can control all the messages sent/received by \( R_j \). \( A \) chooses some random value \( r'_i \in Z_q^* \) and calculates \( Y_1 = r'_i \cdot P \) and \( Y_2 = r'_j \cdot X_1 \cdot P_{pub} \). It then assumes a random \( msk' \) in place of \( msk \) and computes \( C_j' \) as follows:

\[
C_j' = h_A(ID_{U_i}, h_1(msk', ID_{U_i}), ID_{R_j}, Y_1, Y_2, t_{R_j})
\]

Since \( A \) controls the messages being sent out from \( R_j \), therefore \( R_j \) sends \( M_2 = \{Y_1, C'_j, ID_{R_j}, t_{R_j}\} \) to \( U_i \). \( U_i \) on receiving \( M_2 \) calculates \( C_j \) as below:

\[
C_j = h_A(ID_{U_i}, P_{U_i}, ID_{R_j}, Y_1, Y_1 \cdot X_2, t_{R_j})
\]

\( U_i \) finds that the calculated \( C_j \) is not equal to \( C_j' \) received in \( M_2 \) (since \( P_{U_i} \neq h_1(msk', ID_{U_i}) \)) and terminates the authentication process. Therefore, \( A \) cannot successfully forge message \( M_2 \) in this case as well.

2) Replay Attacks: When an attacker \( A \) repeats or delays a message (say \( M_3 \)), it reaches the recipient (i.e., RSU \( R_j \)) at timestamp \( t'_{U_i} \). After receiving the message, \( R_j \) will first verify the validity of the message by checking the freshness of timestamp \( t_{U_i} \), i.e., whether

\[
t'_{U_i} - t_{U_i} \leq \Delta t
\]

if not, then the session is terminated. This makes the protocol resistant to replay attack.

### Table I: A Comparative Summary of Key Features

| Features                  | SACRIFICE | [12] | [7] | [10] | [13] |
|---------------------------|-----------|------|-----|------|------|
| Mutual Authentication     | ✓         | ✓    | ✓   | ✓    | ✓    |
| User Anonymity            | ✓         | ✓    | ✓   | ✓    | ✓    |
| Untraceability            | ✓         | ✓    | ✓   | ✓    | ✓    |
| Non-Reputation            | ✓         | ✓    | ✓   | ✓    | ✓    |
| Unlinkability             | ✓         | ✓    | ✓   | ✓    | ✓    |
| Resistance to Man-in-the-Middle Attacks | ✓ | ✓ | ✓ | ✓ | ✓ |
| Resistance to Replay Attacks | ✓ | ✓ | ✓ | ✓ | ✓ |

Table I shows a comparative summary of SACRIFICE with four state-of-the-art papers [3, 7, 10, 13] on the basis of key security features achieved by these schemes. It is evident that SACRIFICE outperforms all the other schemes considerably.

### V. Performance Analysis

In this section, we evaluate the performance of SACRIFICE both theoretically and experimentally.

#### A. Theoretical Analysis

This section evaluates SACRIFICE in terms of its various overheads. It also analyzes the robustness of the scheme in terms of cracking probability.

1) Overhead Analysis: The computation, communication and storage overheads are measured in terms of execution time, number of transmitting and receiving bytes and number of bytes stored in the memory respectively. Table II summarizes the notations used. During analysis, we consider that the size of each element in \( G \) and \( Z_q^* \) of the elliptic curve is 128 bytes and 20 bytes respectively. We consider the size of timestamp variables (denoted as \( |T| \)) are 4 bytes. The overheads are calculated for a single round of mutual authentication, report generation etc. We also compare SACRIFICE with one competitor scheme [13].

### Table II: Notations used for Theoretical Analysis

| Time taken for | Notation |
|----------------|----------|
| Scalar multiplication | \( T_M \) |
| Bilinear pairing | \( T_{BP} \) |
| Exponentiation | \( T_E \) |
| Hash operation | \( T_H \) |

#### Computation Overhead: Table III shows the computation overhead of SACRIFICE and the competing scheme. It is evident from the Table that the overhead for SACRIFICE is less than its competitor [13] for the first three phases. Additionally, SACRIFICE performs better than its competitor for the last two phases as well because of the absence of expensive operations like exponentiations, bilinear pairings etc.

### Table III: Comparative Analysis of Computation Overhead

| Phase | Entity | SACRIFICE | Competing Scheme [13] |
|-------|--------|-----------|-----------------------|
| Device Registration (Vehicle) | TA/SA | \( 2T_M + 2T_{BP} + T_E \) | \( 2T_M + 2T_{BP} + T_E \) |
| Registration (RSU) | RSU | \( 2T_M + 2T_{BP} + T_E \) | \( 2T_M + 2T_{BP} + T_E \) |
| Mutual Authentication | Vehicle | \( 2T_M + T_E \) | \( 2T_M + T_E \) |
| Report Generation | RSU | \( 2T_M + 2T_E \) | \( 2T_M + 2T_E \) |
| Report Processing | TA/SA | \( 2T_M + 2T_E \) | \( 2T_M + 2T_E \) |
Communication Overhead: Table IV provides communication and storage overhead analysis. From the Table, we observe that the communication overhead for SACRIFICE is significantly less than the overheads of the competitor. This is because SACRIFICE uses lightweight cryptographic tools like hash functions which reduces the size of the tokens exchanged during communication.

Storage Overhead: We observe from Table IV that the storage overhead for SACRIFICE is also less than its competitor. Even though the RSU stores an additional 128 bytes during mutual authentication, the overall storage overhead of SACRIFICE is still less than the work [13].

TABLE IV: Comparative Analysis of Communication and Storage Overheads

| Phases          | Entity          | Communication Overhead (bytes) | Storage Overhead (bytes) |
|-----------------|-----------------|--------------------------------|-------------------------|
|                 | SACRIFICE       | Composed Scheme [13]           | Composed Scheme [13]    |
| Mutual Authentication | Vehicle | [18] 222 222 333 | [18] 722                  |
|                  | RSU             | [18] 114 114 122       | [18] 322                  |
| Report Generation | Vehicle | [18] 222 222 222     | [18] 722                  |
|                  | RSU             | [18] 114 114 114     | [18] 322                  |
| Report Processing | Vehicle | [18] 222 222 222     | [18] 722                  |
|                  | RSU             | [18] 114 114 114     | [18] 322                  |
|                  | AA/RA           | [18] 114 114 114     | [18] 322                  |
|                  | CS              | [18] 114 114 114     | [18] 322                  |

Summarily, we observe from the entire overhead analysis that SACRIFICE is less intensive in terms of computation, communication and storage overheads than its competitor. This is because a less intensive mathematical approach is used in our proposed scheme. However, this approach maybe vulnerable to brute-force attacks, hence we explore the probability of breaking SACRIFICE by a brute-force attack in the following subsection.

2) Cracking Probability: Cracking probability is defined as the probability of cracking a token while it is being transmitted or stored by an adversary [18]. Mutual Authentication is a crucial step in SACRIFICE. If an attacker breaks this step, it can gain control over the network and its confidential information. It can also send malicious data to other participants by compromising them. The following cases describe the probability of cracking the mutual authentication algorithm when an adversary launches a brute-force attack on the network.

Case 1: When an attacker A attempts to generate message $M_1 = \{X_1, X_3, C_i, t_{U_i}\}$, by posing as an authentic vehicle in the network the following may happen:

The tokens $X_1, X_3$ can be easily calculated by A with any random number $r_i \in Z_q^*$ and some identity ID$_{U_i}$. However, for $C_i$, it does not have the value ID$_{U_i}$ (since the vehicle is not registered). Therefore, A has to select a value for $C_i$. The length of the token $|C_i| = |Z_q^*| = n$ bits (Say).

Total possible combinations of the $C_i$ bits $= 2^n$

Probability that the correct combination for this case is selected $= \frac{1}{2^n}$

Therefore, the cracking probability (%) $= \frac{1}{2^n} \times 100$

For example, for $|Z_q^*| = 64$ bits, the cracking probability (%) $= 5.4 \times 10^{-18}$ is very low.

Case 2: When an attacker A attempts to generate message $M_2 = \{Y_1, C_j, t_{R_j}\}$, by posing as an authentic RSU in the network, the following may occur:

The tokens $Y_1$ can be easily calculated by A with any random number $r_j \in Z_q^*$. However, it does not have the necessary tokens to calculate the value of $C_j$. Therefore, A has to select a value for $C_j$ and similar to Case 1, the Cracking Probability (%) $= \frac{1}{2^n} \times 100$.

Thus, it is clear from the above discussion that due to the very low cracking probability, an attacker A cannot infiltrate the system by a brute-force attack within the short span of time that a particular vehicle stays in the range of an RSU.

B. Experimental Evaluation via Simulation

Here, we implement a prototype of SACRIFICE and validate its performance with a state-of-the-art competitor [13].

1) Simulation Environment: We implement SACRIFICE and its competitor with the help of two simulators: SUMO and NS3. For both the schemes, the pairing and group related operations are performed using the popular PBC Library [19] of NS-3, where we have used Type A pairings based on the elliptic curve, $y^2 = x^3 + x$. Table V summarizes the simulation parameters used in our setup. The simulation scenario in SUMO consists of a single street, 1000 m long, with 2 lanes shown in Fig. 3a. The RSUs are deployed at an equal distance of 200 m from each other to provide maximum coverage. Fig. 3b shows a snapshot of the simulation in NS3 taken at a particular time, $t = 32.709$ seconds. It shows 42 vehicles in the network (represented as red circles) which interact with the RSUs (represented as blue circles). It can be seen that some vehicles are interacting with the RSUs while a few have moved out of scope of all RSUs and others are still to enter the scope of any RSU. It is also observed that an RSU is sending a final road condition report to the CS which then forwards it to the Application Authority.

![Fig. 3: Simulation Snapshots](image)

TABLE V: Simulation Parameters

| Parameters                | Value                           |
|---------------------------|---------------------------------|
| Area (SUMO)               | 200 x 1000 m$^2$                |
| Duration (NS-3)           | 5 minutes                       |
| Range of Entities (NS-3)  | 300 meters                      |
| Wireless Protocols (NS-3) | 802.11p for message transmission|
|                          | 802.11b for beacon broadcasting |
The following two metrics are used in evaluating such network performance:

**End-to-end Delay**: Delay is defined as the average time taken since the moment a vehicle starts transmitting the packet until its successful delivery \[20\]. In our work, we consider the end-to-end delay as the average delay of all the packets transmitted in the network within the simulation duration.

\[
\text{Avg. Delay} = \frac{\text{sum of delay of all packets received}}{\text{total no. of packets received}}
\]

**Packet Delivery Ratio (PDR)**: It is measured as the ratio of the total number of packets delivered to the destination to the total number of packets sent to the destination over a period of time \[21\], \[22\].

\[
PDR = \frac{\text{total no. of packets delivered}}{\text{total no. of packets sent}}
\]

### 3) Results and Discussion

We conduct four sets of experiments to evaluate the performance of SACRIFICE and its competitor scheme \[13\]. In the simulation, vehicles start to send their own authentication messages after they receive the beacon broadcast from the nearest RSU. Such broadcasts happen every 0.3 milliseconds. An average result of 10 independent runs is taken while plotting the simulation graphs.

In the first set of experiment, we plot (Fig. 4) the execution time of the various sub-tasks. We observe that the time taken by each sub-task in SACRIFICE is substantially less compared to its state-of-the-art competitor \[13\]. SACRIFICE roughly takes around 80.96% on average less execution time for each of the sub-tasks compared to its competitor scheme.

In the second set of experiment, we observe (Table VI) the time taken by each entity in SACRIFICE is less compared to that of its competitor. We also observe that the time taken is of the order of a few nanoseconds and hence is feasible for time-critical VANET applications. As explained earlier, the reason that SACRIFICE performs better in both these sets of experiments is because it uses less intensive mathematical operations as compared to its competitor.

**TABLE VI: Average Time taken by each Entity**

| Entity     | SACRIFICE (ns) | Competing Scheme \[13\] (ns) |
|------------|----------------|-----------------------------|
| Vehicle    | 3.552          | 34.176                      |
| RSU        | 9.31           | 28.13                       |
| CS         | 0.58           | 9.74                        |
| AA/RA      | 0.596          | 3.65                        |

In the third set of experiment, we plot (Fig. 5) the average end-to-end delay with increasing number of vehicles in the network. From the figure, it is evident that the average delay increases when the number of vehicles increases in the network. The increase in the number of vehicles results in increased number of packet transmission resulting in increased congestion. This network congestion, in turn, increases the end-to-end delay of each packet. On an average, the delay for SACRIFICE shows 37.5% better performance as compared to the work \[13\].

In the fourth set of experiment, we plot (Fig. 6) the packet delivery ratio (%) with increasing number of vehicles in the network. We observe from the figure that for both the schemes there is a negative trend in the graph, i.e. PDR decreases with the increasing number of vehicles in the network, as expected. However, the ratio is stable and stays above 97% for both the schemes. We also observe that as the number of vehicles approaches 90, the PDR for the proposed scheme starts improving compared to its competitor. Thus, when the number of vehicles in the network is large, SACRIFICE scales well or performs better.

### VI. Conclusion

We propose a secure road condition monitoring scheme SACRIFICE over fog-based VANET, which is low-overhead and scalable. In this scheme, whenever a vehicle encounters a bad road condition (e.g. accident) in the network, it sends a report to the closest RSU only after performing mutual authentication between vehicle and RSU. The RSU then generates the final report and sends it to the CS for further processing. Apart from maintaining the important security features like mutual authentication, user anonymity, the scheme also ensures additional security features like non-repudiation, un-linkability and untraceability. The detailed analysis of the security features shows that our scheme is robust against both external and internal adversaries. Performance of the scheme is evaluated both through theoretical overhead analysis and simulation using SUMO and NS-3 platform to show its viability for practical implementation. The overhead analysis shows our scheme’s dominance over a state-of-the-art competitor. Simulation results also corroborate the theoretical analysis in terms of execution time while achieving better network performance in terms of end-to-end delay thereby establishing its applicability in time-sensitive VANET application. In future, the scheme may be extended by including vehicle-to-everything (V2X) communications and its associated security issues. Introducing the concept of scheduling to improve the overall latency further is another open research area.
REFERENCES

[1] Xiaodong Lin, Rongxing Lu, Chenxi Zhang, Haojin Zhu, Pin-han Ho, and Xuemin Shen. Security in vehicular ad hoc networks. *IEEE Communications Magazine*, 46(4):88–95, 2008.

[2] J. Sengupta, S. Ruj, and S. D. Bit. A Secure Fog-Based Architecture for Industrial Internet of Things and Industry 4.0. *IEEE Transactions on Industrial Informatics*, 17(4):2316–2324, 2021.

[3] Mingming Cui, Dezhi Han, and Jun Wang. An Efficient and Safe Road Condition Monitoring Authentication Scheme Based on Fog Computing. *IEEE Internet of Things Journal*, 6(5):9076–9084, 2019.

[4] Hassan Ali Khattak, Saif Ul Islam, Ikram Ud Din, and Mohsen Guizani. Integrating Fog Computing with VANETs: A Consumer Perspective. *IEEE Communications Standards Magazine*, 3(1):19–25, 2019.

[5] Luobing Dong, Qifen Ni, Weili Wu, Chuanhe Huang, Taibz Znati, and Ding Zhu Du. A Proactive Reliable Mechanism-Based Vehicular Fog Computing Network. *IEEE Internet of Things Journal*, 7(12):11893–11907, 2020.

[6] Sultan Basudan, Xiaodong Lin, and Karthik Sankaranarayanan. A Privacy-Preserving Vehicular Crowdsensing-Based Road Surface Condition Monitoring System Using Fog Computing. *IEEE Internet of Things Journal*, 4(3):772–782, 2017.

[7] Mimi Ma, Debiao He, Huaqun Wang, Neeraj Kumar, and Kim-Kwang Raymond Choo. An Efficient and Provable Secure Authenticated Key Agreement Protocol for Fog-Based Vehicular Ad-Hoc Networks. *IEEE Internet of Things Journal*, 6(5):8065–8075, 2019.

[8] Maryam Rajabzadeh Asaar, Mahmoud Salmasizadeh, Willy Susilo, and Akbar Majidi. A Secure and Efficient Authentication Technique for Vehicular Ad-Hoc Networks. *IEEE Transactions on Vehicular Technology*, 67(6):5409–5423, 2018.

[9] N. B. Gayathri, Gowri Thumbur, P. Vasudeva Reddy, and Muhammad Zia Ur Rahman. Efficient Pairing-Free Certificateless Authentication Scheme With Batch Verification for Vehicular Ad-Hoc Networks. *IEEE Access*, 6:31808–31819, 2018.

[10] Xiong Li, Tian Liu, Mohammad S. Obaidat, Fan Wu, Pandi Vijayakumar, and Neeraj Kumar. A Lightweight Privacy-Preserving Authentication Protocol for VANETs. *IEEE Systems Journal*, 14(3):3547–3557, 2020.

[11] SK Hafizul Islam, Mohammad S. Obaidat, Pandi Vijayakumar, Enas Abdullhay, Fagen Li, and M Krishna Chaitanya Reddy. A robust and efficient password-based conditional privacy preserving authentication and group-key agreement protocol for VANETs. *Future Generation Computer Systems*, 84:216–227, 2018.

[12] Jie Cui, Xuefei Tao, Jing Zhang, Yan Xu, and Hong Zhong. Hcpa-gka: A hash function-based conditional privacy-preserving authentication and group-key agreement scheme for vanets. *Vehicular Communications*, 14:15–25, 2018.

[13] Yujue Wang, Yong Ding, Qianhong Wu, Yongzhuan Wei, Bo Qin, and Huojong Wang. Privacy-Preserving Cloud-Based Road Condition Monitoring With Source Authentication in VANETs. *IEEE Transactions on Information Forensics and Security*, 14(7):1779–1790, 2019.

[14] Rajib Banerjee and Sipra Das Bit. An energy efficient image compression scheme for wireless multimedia sensor network using curve fitting technique. *Wireless Networks*, 25(1):167–183, 2019.

[15] Amrita Ghosal, Subir Halder, Sanjib Sur, Avishek Dan, and Sipra Das Bit. Ensuring Basic Security and Preventing Replay Attack in a Query Processing Application Domain in WSN. In *Computational Science and Its Applications – ICCSA 2010*, pages 321–335, 2010.

[16] P. Banerjee, N. Nikam, and S. Ruj. Blockchain Enabled Privacy Preserving Data Audit. *CoRR*, abs/1904.12362, 2019.

[17] J. Sengupta, S. Ruj, and S. Das Bit. An Efficient and Secure Directed Diffusion in Industrial Wireless Sensor Networks. In *Proceedings of the 1st International Workshop on Future Industrial Communication Networks (FICN), MobICom*, page 41–46, 2018.

[18] Amrita Ghosal, Subir Halder, Sanjib Sur, Avishek Dan, and Sipra Das Bit. Ensuring Basic Security and Preventing Replay Attack in a Query Processing Application Domain in WSN. In *Computational Science and Its Applications – ICCSA 2010*, pages 321–335, 2010.

[19] Ben Lynn. *PBC Library* : The Pairing-Based Cryptography Library. https://crypto.stanford.edu/pbc/, 2013.

[20] Jiawei Huang, Qi Li, Shaohua Zhong, Lianhai Liu, Ping Zhong, Jianxin Wang, and Jin Ye. Synthesizing Existing CSMA and TDMA Based MAC Protocols for VANETs. *Sensors*, 17(2), 2017.

[21] Kunal Vyas, Jayasree Sengupta, and Sipra Das Bit. ARPL: Supporting Adaptive Mixing of RPL Modes to Overcome Memory Overflow. In *2018 IEEE International Symposium on Smart Electronic Systems (iSES) (Formerly iNiS)*, pages 124–129, 2018.