A Location-Aware and Healing Attestation Scheme for Air-Supported Internet of Vehicles

El-Zawawy, Mohamed A.; Lal, Chhagan; Conti, Mauro

DOI
10.1109/TITS.2023.3316775

Publication date
2023

Document Version
Final published version

Published in
IEEE Transactions on Intelligent Transportation Systems

A Location-Aware and Healing Attestation Scheme for Air-Supported Internet of Vehicles

Mohamed A. El-Zawawy, Chhagan Lal, and Mauro Conti, *Fellow, IEEE*

***Abstract*— With the rapid technological advancement in the Internet of Things (IoT) and Internet of Vehicles (IoV), we witness exponential growth of Connected and Autonomous Vehicles (CAVs). However, these integrations of IoV with other technologies make the IoV network and its interaction between different network components highly complex. Therefore, ensuring the correct functioning of the firmware and software running on these next-generation vehicles becomes an essential requirement. A feasible method to address the aforementioned security issues is Remote Attestation (RA). However, the advancement in the attackers' approaches and the increased complexity, large network size, and vehicle mobility allow the attacks to bypass these security solutions, making RA less effective. In this paper, we propose LHASIoV, an attestation and healing protocol for IoV. LHASIoV has many features such as competent-wise (treats different entities of the system differently), geographical location-aware (traces forensics of security breaches and eases healing compromised vehicles), gradual healing (via slicing the healing software) of compromised vehicles, and resistance to single-point-of-failure. We provide proof-of-concept implementation and formal operational and security analysis for LHASIoV. To show its practical feasibility and effectiveness, we provide performance analysis by implementing it on the Omnetpp simulator. The simulation results show that for an IoV system that has 100 vehicles moving with a speed range of 15-25 mph, LHASIoV needed only 5.27 seconds to complete the vehicle's attestation. For this number of vehicles and compared to the existing protocols, LHASIoV reduced the communication and storage costs on average by 54.46% and 43.92%, respectively.**

***Index Terms*— The Internet of Vehicles, drones, attestation, air-supported networks, operational semantics.**

I. Introduction

VEHICULAR Ad Hoc Networks (VANETs) are wireless networks of vehicles. Internet of Vehicles (IoV) can be realized as advanced vehicular networks benefiting from and combining Vehicular Ad Hoc Networks (VANETs) and Internet of Things (IoT). There is a growing adoption of IoV in the vehicular industry. This is naturally associated with an increasing number of autonomous and connected vehicles. Intelligent transportation technology (e.g., automated driving) is one of the applications of IoV. Securing a huge number of internet-connected vehicles becomes an interesting research area [1]. Every vehicle has 70 to 100 Electronic Control Units (ECUs) running firmware that gets regularly updated. This adds to the complexity and hence to the challenge of securing these automotive electronics. The continuous movement of vehicles creates a need for an attestation protocol for verifying their integrity during their transit [2], [3].

A ground network solely is not able to fulfill needs and support the demand of IoV [4], [5]. This is confirmed by frequent accidents that occur to vehicles applying self-driving technology. A ground network in this context is challenged by many issues including unlimited connections to the high mobility of vehicles, absence of network coverage (in rural areas), and coverage interruption (in mountainous areas) [6]. This becomes more evident when we recall that vehicles communicate (in 5.9 Ghz to 75 MHz band) with infrastructure and with each other via Dedicated Short-Range Communication (DSRC) [7], [8]. The Air-Supported Internet of Vehicles Networks (ASIoVNs) solve this ground network problem [9]. The flying objects (such as drones and balloons) in ASIoVNs support the ground network. When necessary (for example in remote areas), flying objects (e.g., drones and balloons) can provide processing, data acquisition, seamless connection, and transmission services [10]. However, the security of ASIoVNs is not mature enough, as confirmed by recent security incidents [11], [12].

The technology of remote attestation is convenient to ensure the security, safety, integrity, and credibility of different entities in ASIoVNs including ECUs running on vehicles. Network attestation can be defined as the process of witnessing the signing of genuine network software. The process also involves signing this software to ensure that it is conveniently treated by those who rely on its functionality. Therefore, network attestation can be realized as a formal acknowledgment of the authenticity of network software and verification of its proper treatment. Network attestation is mainly achieved using general security tools including communicated encrypted and/or signed messages. Although few protocols were proposed to attesting ASIoVNs [6], these protocols suffer from various issues such as:

1) Not attesting different components in the system according to their nature. This is so because rather than

Manuscript received 2 February 2023; revised 28 June 2023; accepted 13 September 2023. Date of publication 2 October 2023; date of current version 2 February 2024. This work was supported in part by Project SERICS through the NRRP MUR program funded by the EU - NGEU under Grant PE00000014. The Associate Editor for this article was M. H. Anisi. *(Corresponding author: Mohamed A. El-Zawawy.)*

Mohamed A. El-Zawawy is with the Department of Mathematics, Faculty of Science, Cairo University, Giza 12613, Egypt (e-mail: maelzawawy@cu.edu.eg).

Chhagan Lal is with the Department of Intelligent Systems, Delft University of Technology (TU Delft), 2628 CN Delft, The Netherlands (e-mail: chhagan.iiita@gmail.com).

Mauro Conti is with the Department of Mathematics, and the HIT Research Center, University of Padua, 35131 Padua, Italy, also with the Faculty of Electrical Engineering, Mathematics, and Computer Science, Delft University of Technology (TU Delft), 2628 CN Delft, The Netherlands, and also with the Department of Electrical and Computer Engineering, University of Washington, Seattle, WA 98195 USA (e-mail: mauro.conti@unipd.it).

Digital Object Identifier 10.1109/TITS.2023.3316775

vehicles, ASIoVNs hosts static entities and flying objects.
2) Geographical locations of vehicles at attestation time are not typically reported. Such locations can provide evidence when tracing any security breaches and when healing compromised vehicles.
3) Suffering from single-point-of-failure.
4) Not supported with formal semantics to reason formally about protocol steps.

To overcome these issues, for ASIoVNs we propose a new attestation and healing protocol, called LHASIoV (Location-aware and Healing attestation scheme for Air-Supported Internet of Vehicles).

LHASIoV is composed of two phases. The first phase is the registration and key generation which is carried out by the Trusted Authority (TA) in a secured environment to deploy the network devices. The second phase, attestation, is initiated repeatedly by TA to check the integrity of software managing devices of the network. The attestation phase also heals infected vehicles of the network. The attestation phase executes three types of attestations carried in the following order: side units attestation, flying objects attestation, and vehicle attestation. Each type builds upon and benefits from the results of its previous one. However, these attestation types are repeated with different frequencies as their attested objects have different threats, importance, and nature.

We implemented and assessed LHASIoV performance from different perspectives. We make available result files from the simulations tool we employed to assess LHASIoV in a public repository.[1] In this paper, we introduce a proof-of-concept implementation for LHASIoV. This implementation is based on the security architecture named TrustLite which in turn is built on a Siskiyou Peak [13] owned by *Intel*. We carried multi-facets comparisons of LHASIoV against related state-of-the-art schemes [6], [14], [15]. The comparisons prove that LHASIoV is superior when compared to related state-of-the-art protocols. We also proved the practical feasibility of LHASIoV. We did this by implementing LHASIoV in Omnetpp, a popular networking simulation tool, associated with the Castalia simulator. For an IoV system that has 100 vehicles that are moving with a speed range of $15-25$ mph, LHASIoV needed only 5.27 seconds to complete the vehicle's attestation. For this number of vehicles and compared to the related state-of-the-art protocols, LHASIoV reduced the communication and storage costs on average by 54.46% and 43.92%, respectively. One may have the concern that for some emergency scenarios in autonomous vehicles, 5.27 seconds might look long. This concern disappears if we recall that LHASIoV is an attestation protocol (not a routing one) and compared to state-of-the-art, this time is an improvement over currently achievable. Besides, attestation techniques are executed occasionally and do not directly contribute to making critical driving decisions.

a) Contribution: Compared to the related state-of-the-art schemes, in a nutshell, the following are the main advantages of LHASIoV:

1) To the best of our knowledge, LHASIoV is the first attestation protocol to be supported with precise operational semantics that can be used to reason about the protocol and prove its security.
2) LHASIoV has no single point of failure: every phase of LHASIoV relies on secure components of the previous phase as trusted entities. LHASIoV has many provers to reduce overhead and avoid single-point-of-failure. This enables partial collective attestation and hence distributes the attestation calculations to many parties. It is worth noting that LHASIoV is not fully collective as not all system entities act as provers.
3) LHASIoV is competent-wise: it treats different entities of the system differently according to their characteristics (such as hardware).
4) LHASIoV is geographical location-aware. This enables using LHASIoV for tracing forensics of any security breaches and eases the process of healing compromised vehicles.
5) LHASIoV heals gradually (via slicing the healing software) compromised vehicles.

b) Organization: The remainder of the paper is organized as follows. The related work is outlined in Section II. Section III presents the research framework of this paper. This involves introducing the network model, threat model and problem statement, and requirements in Sections III-A, III-B, and III-C, respectively. Section IV introduces LHASIoV in details. In Section V, operational semantics and security analysis to LHASIoV are introduced. The evaluation to LHASIoV performance and discussion are presented in Section VI. Section VII concludes the paper and provides directions for future work.

II. Related Work

In this section, we review the state-of-the-art attestation schemes proposed for providing security and privacy in various IoV scenarios [14], [16], [17], [18], [19], [20].

Authors in [15] propose an approach to provide privacy-preserving communications between vehicles and their embedded sensors in VANETs. First, the authors identified several challenges and weaknesses in the existing solutions (e.g., changing pseudonyms, and using revocation policies to remove malicious nodes) that support communication privacy in VANETs. Therefore, to enhance the privacy of the state-of-the-art, the paper proposes using trusted computing techniques for V2X scenarios. In particular, a decentralized approach is proposed, leveraging anonymous credentials through Direct Anonymous Attestation (DAA). The authors show that their proposed solution is scalable and decentralized. It is the first that applies the DAA algorithms to provide strong privacy protection and user-controlled linkability without the limitations of current pseudonym schemes. Moreover, it proposes simpler DAA-based versions of pseudonym provision, management, and revocation, only needing a limited set of infrastructure entities, and efficiently pulls misbehaving vehicles without disclosing the vehicle's identity or needing heavy computational technologies. For dynamic networks, Kohnhäuser et al. [14]

[1] https://github.com/maelzawawy/LHASIoV

introduced SALAD, a collective attestation protocol. SALAD achieves well in highly disruptive and dynamic topologies. Although these attributes make SALAD applicable for IoV systems, it does not consider many modern features of IoV systems (such as static components).

Since it is important to authenticate a device before it can be attested by an attesting or verifying entity. The authors in [3] propose a security scheme using Physically Unclonable Functions (PUFs) to perform a combined authentication and attestation in IoV scenarios. The OBUs are first authenticated by the RSUs, and then the edge servers will carry out the attestation procedure for them. As it is not practical for RSUs to perform large-scale attestations due to resource constraints, an edge server co-located with the RSU is used for the verification process.

To avoid the need for attestation hardware (i.e., Trusted Platform Module (TPM)) at Electronic Control Units (ECUs) of intelligent vehicles, and to ensure the safe and secure operation of ECUs in vehicles, authors in [21] propose an attestation scheme that primarily targets the safety of vehicles. In their scheme, whenever the vehicle is unlocked and its door is opened, a trusted main ECU checks the software integrity of the vehicle's safety-critical ECUs. The main ECU ensures that the engine starts only after the successful verification of all the safety-critical ECUs. Moreover, the proposed scheme consists of two attestation methods to address the heterogeneity of the ECUs in vehicles. The first attestation method is for simple ECUs that possess only a small and modest system architecture (e.g., basic sensors and actuators), while the second method targets advanced ECUs (such as Advanced RISC Machine (ARM) application processors) that are used for more complex tasks. The paper shows that the proposed scheme has low deployment cost as both the attestation methods are applicable to a broad range of existing commodity ECUs.

Recently, the usage of space–air–ground integrated networks (SAGIN) in rural and remote areas has increased to support seamless connection services. By leveraging terrestrial networks, satellite networks, and air networks, it can better support car networking data acquisition, processing, and transmission, thereby improving the vehicle driving experience, which continues to promote the development of intelligent transport. Nevertheless, few recent security incidents have prevented the rapid development of the SAGIN-based IoVs. Authors in [6] propose an attestation scheme in the virtual machine (VM) environment to efficiently collect and verify the trusted information. To perform their proposed attestation procedure, first, they study the VM's trust chain approach in which the critical components are measured to be extended to the platform configuration register. Next, the VM monitor collects the trusted information of the VM's trust chain by batch through the VM introspection (VMI) technique. Then, the TPM is used to encrypt and protect the collected information. At last, the VMs securely transmit the collected trusted information to the attestation service at the cloud management center, where it is analyzed and stored. The attestation results are also displayed at the cloud management center.

Kim et al. [18] introduced a cooperative infrastructure for aerial-ground systems for surveillance and epidemic prevention using AI-supported communications. For IoT-assisted maritime transportation systems, Lee et al. [19] presented a framework to build differential security barriers to detect virtual emotion. For infrastructure-less IoV systems, Tan et al. [20] presented an authentication mechanism to facilitate data transmission. To avoid active-edge IoV infrastructure, their work relies on the tethered UAV (TUAV) as a mobilized base station. Unfortunately, these methods are not conveniently applicable for attesting modern IoV systems.

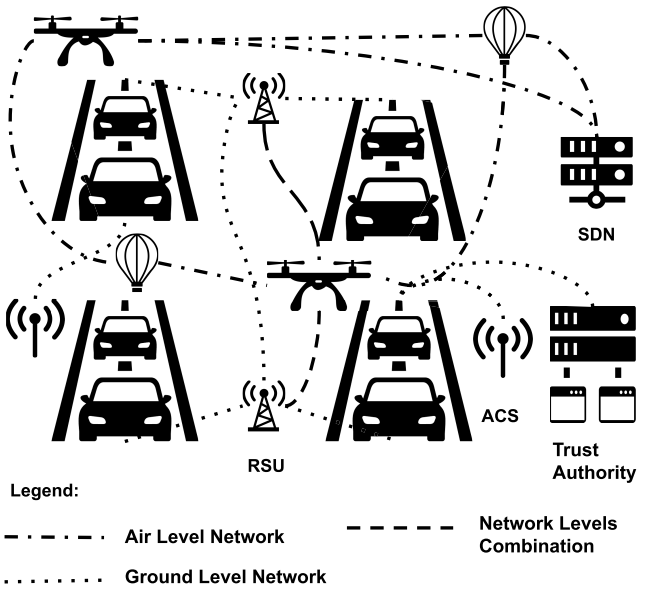

Fig. 1. The ASIoVNs model for IoV studied in this paper.

III. Research Framework

This section presents the research framework of this paper. This involves introducing the network model, threat model and problem statement, and requirements in Sections III-A, III-B, and III-C, respectively.

A. Network Model

Fig 1 presents the IoV model of ASIoVNs studied in this paper. Our network model has two levels: air and ground levels. The air level of the network is a network of drones and balloons. The air level is typically supported with Software Defined Networks (SDNs). The job of SDNs is to control the behavior of air level and to optimize the use of network resources. The software of SDNs is stored on cloud computing or servers which are local or remote according to the technology of the used SDNs. Both levels of the network communicate with each other and enable nodes to join and exit the system.

The ground level of the network has vehicles, Road Side Units (RSUs), TA, and Access Control Servers (ACSs). Mainly, network services and mechanisms are meant to support vehicles. RSUs are static network entities distributed on the roadsides and are essential for achieving collaborative distributed management in IoV. In particular, RSUs participate in routing delays and data transmission, (like data dissemination,

traffic directory, and security management). RSUs can also act as switches for SDNs. This helps efficiently update flow rules that are affected by the high mobility of vehicles and hence the frequent changes in the network topology. ACSs [22] are network components that provide certain IoV-services (such as speed control). Therefore, ACSs guarantees optimal and stable performance in the network. Generally speaking, RSUs are more essential entities than ACSs in IoV-networks. Although the role of RSUs is typically the same in different applications of IoV-networks, this is not true for the role of ACSs. The servers of SDNs support the computing and connectivity of the model. Except for ACSs which has the privilege of getting extra resources from servers, the computational power of all system entities (i.e., vehicles and RSUs) is limited. The TA is responsible for registering network entities and doing all necessary attestation auxiliary actions. In this paper, we refer to RSUs and ACSs as Side Unites (SUs) and to drones and balloons as Flying Objects (FOs).

Our network model has special characteristics that make it special and hinder treating the model with traditional attestation approaches. The model has a mix of static and highly dynamic entities. Also, the functionality, role, and structure of different entities put them at different levels of security risk. Hence, they need to be attested with different strategies and at different frequencies. For example, attesting\healing infected software of static entities (like RSUs) is a simpler problem than attesting\healing a highly dynamic vehicle whose geographical location is unknown. What makes the problem even harder than usual is that both RSUs and vehicles interact and collaborate within the same system.

B. Threat Model and Problem Statement

In some societies, IoV can have high economic value, and hence security threats for IoV can influence the economy of these societies. Therefore, unduly attackers are lured to target IoV networks. As a wide-range service-providing network, our network model may be attacked by malicious users while communicating system messages and providing system services. This can be done by changing the commands of system services and software for profit. Moreover, attackers may change the addresses of attestation protocols, eavesdrop on system communications, and send fake messages. Hence attackers can execute man-in-the-middle attacks on the system. Rather than affecting the integrity, confidentiality, and feasibility of the network, such attacks can result in massive overhead for the system.

This paper adopts a widely accepted threat model, namely *honest-but-curious* (HBC) [6]. We assume that all system parties are benign and execute the attestation protocol properly. One sort of privacy leakage that is assumed in our model can be done by malicious ACSs reading sensitive vehicle information. Malicious vehicles may try to get away with not paying for a used service. To execute an escape attack, malicious vehicles may destroy running sessions of ACSs. Our choice of the HBC adversary model is due to the privacy properties of IoV systems for which unlinkability and undetectability of messages are desirable characteristics. Moreover, privacy and anonymity of IoV systems rely on these characteristics. Other older models such as DY [23] adversary model benefit from the notions of observational equivalence and indistinguishability. Although, for realistic IoV systems, the DY model is too strong to be used, we analyze the security characteristics of `LHASIoV` using the DY model.

All in all, the problem treated in this paper is to develop a protocol that securely and remotely attests to the integrity of the state of software running on IoV system entities of our network model. As the consequences of hardware failure of flying system entities (drones and balloons) are more critical than those of other system entities, the required protocol should also attest to the hardware functionality of these special entities. Such protocol hence enables revealing potential malware on different system entities. The other goal of this protocol is to disinfect (heal) infected vehicles. The healing process involves restoring benign states of vehicle software.

C. Requirements

This paper aims at developing a protocol for IoV attestation to guarantee the trustworthiness of IoV using our network model. More specifically, the proposed solution must satisfy the following requirements:

1) Ensuring trustworthiness of our IoV network model against the threats presented above including man-in-the-middle and redirection attacks. This can be achieved by a protocol that prevents updates to attestation steps and software of network entities. Also, the protocol should be able to stop an adversary from utilizing self-created data to interact with a challenge.
2) Considering the highly dynamic aspects and preserving the all-time stability of IoV. This can be satisfied by a protocol that is minimal in its communication, computational, and storage needs, compared to state-of-the-art protocols.
3) Healing infected vehicles. This can be done by reestablishing the software of infected vehicles into benign states.
4) Having no single point of failure. This can be done by relying on different trusted entities of the system.
5) Reporting geographical location-aware results for vehicle attestation.

IV. Novel Attestation Protocol

The attestation protocol proposed in this paper, named `LHASIoV`, is composed of two phases. The first phase is the registration and key generation one (Section IV-A) which is carried by the TA in a secured environment to deploy the network devices. The second phase, attestation one (Section IV-B), is initiated repeatedly by TA to check the integrity of software managing devices of the network. The attestation phase also heals infected vehicles of the network. The notations used in the paper are presented in Table I and the protocol phases are explained in detail in this section.

A. Registration and Key Generation

TA takes care of registering and creating pairs of private and public keys for system entities. For instance, this is done

TABLE I
NOTATIONS USED IN THIS PAPER

Notation	Meaning
$\mathcal{I}_v$	Vehicle ID.
$\mathcal{V}_i$	Intermediate calculation value.
$h()$	Hash function.
$\mathtt{SK}_i$	Private key of of system entity i.
$\mathtt{PK}_i$	Public key of system entity i.
$\mathcal{B}_p(x,y)$	Bi-variate symmetric polynomial.
$GF(p)$	Galois finite field over a sufficiently large prime number p.
Z_p	$\{0,1,\ldots,p-1\}$.
θ	Element of Z_p.
$\|$	Concatenation operation.
`TimeStamp()`	Method that returns the current timestamp of the system.
L	Memory attestation value.
(I^x, O^x)	Valid input and its output for a side unit SU_j^x.
$\mathtt{msg}_i$	Communicated message.
δ	Message signature.
δ^e	Encrypted message.
$\mathtt{Enc}(\mathtt{PK}_{SU_i}, \mathtt{msg}_i)$	Method encrypting the message $\mathtt{msg}_i$ using the key $\mathtt{PK}_{SU_i}$.
$\mathtt{sig}(\mathtt{SK}_{SU_i}, \mathtt{msg}_i)$	Method signing the message $\mathtt{msg}_i$ using the key $\mathtt{SK}_{SU_i}$.
$\mathtt{Int}_i$	Boolean indicating the integrity of the system entity i.
H	Hardware component of a flying object.
(T, R)	Valid hardware test T and its result R for H.
$\mathcal{A}$	Set of side units and flying object agents.
Adv	A software\hardware adversary.
Adv_s	Software adversary.
Adv_h	Hardware adversary.
Aid	ID of vehicle attestation process.

as follows for a vehicle v. TA fixes a unique ID, $\mathcal{I}_v$, and calculates auxiliary value $\mathcal{V}_1 = h(\mathcal{I}_v \parallel \mathtt{SK}_{\mathtt{TA}})$, where h is a hash function. Then TA chooses a bi-variate symmetric polynomial:

$$\mathcal{B}_p(x, y) = \sum_{i,j=0}^{d} c_{i,j} x^i y^j, \tag{1}$$

The polynomial is over a Galois finite field, $GF[x, y](p) = Z_p = \{0, 1, \ldots, p-1\}$, where p is a sufficiently large prime number and the degree of polynomial t is also sufficiently large. TA then selects a nonce $\theta \in Z_p$ and calculates $\mathcal{V}_2 = \mathcal{B}_p(\mathcal{V}_1, \mathcal{I}_v \parallel \theta)$ [24], [25], [26]. The use of a nonce (a random number) in our proposed protocol improves its security and resilience to security attacks as explained later in the paper, specifically in the security analysis section. Afterward, TA uses the auxiliary credentials $\mathcal{V}_1$ and $\mathcal{V}_2$ to calculate keys of v before deploying the vehicle as follows:

$$\mathtt{SK}_{\mathrm{v}} = h((\mathtt{TimeStamp()} \parallel \mathcal{V}_1) \parallel h(\mathcal{V}_1)). \tag{2}$$

and $\mathtt{PK}_{\mathrm{v}} = (p-1) * \mathtt{SK}_{\mathrm{v}}$.

B. Attestation Phase

The second phase of LHASIoV is the attestation one. There are three types of attestation processes carried out in this phase:

1) Type 1: Side units (RSUs and ACSs) attestation.
2) Type 2: Flying objects (drones and balloons) attestation: this type relies on the result of the first type. The methodology of this attestation type enables carrying the attestation in different geographical locations of vehicles. This also enables attesting communications carried out in these locations.
3) Type 3: Vehicle attestation: this type relies on the results of types 1 and 2.

The frequency of carrying Type 3 attestation is typically higher than that of Type 2 which is in turn higher than that of Type 1.

1) Side Units Attestation (Type 1): We present the details of attesting RSUs and ACSs in this section. Periodically (using predefined intervals), each one of the SUs plays the role of a verifier to attest its SUs neighbors which play the role of provers in this case. Algorithm 1 formalizes the steps of this attestation type. The algorithm supposes a verifier SU_i that has k SUs-neighbors, $SU_j^1, \ldots, SU_j^k$. Then, the attestation starts

Algorithm 1 Side Unites (RSUs and ACSs) Attestation.

procedure $SU_i^1()$
  Start the interval timer of SU_i;
  $T \leftarrow \mathtt{TimeStamp}()$;
  for each $x \in 1, 2, \ldots, k$ **do**
    $N^x \leftarrow \{0,1\}^n$;
    $S^x \leftarrow$ Fix a service provided by SU_j^x;
    $(I^x, O^x) \leftarrow$ Fix a valid input and its output for SU_j^x;
    $L^x \leftarrow$ Valid memory attestation of SU_j^x;
    $\mathtt{msg}_i \leftarrow N^x \parallel S^x \parallel I^x$;
    $\delta_i \leftarrow \mathtt{sig}(\mathtt{SK}_{SU_i}, \mathtt{msg}_i)$;
    $\mathtt{Challenge}(\mathtt{msg}_i, \delta_i, SU_j^x)$;
procedure $SU_j^{x1}(msg_i, \delta_i)$
  if $(\mathtt{VerSig}(\mathtt{PK}_{SU_i}; \mathtt{msg}_i, \delta_i))$ **then**
    $N^x \parallel S^x \parallel I^x \leftarrow \mathtt{msg}_i$;
    $O^{x\prime} \leftarrow$ Run the service S^x on I^x;
    $L^{x\prime} \leftarrow \mathtt{MemoryAttest}(SU_j^x)$;
    $\mathtt{msg}'_j \leftarrow N^x \parallel O^{x\prime} \parallel L^{x\prime}$;
    $\delta_j^e \leftarrow \mathtt{Enc}(\mathtt{PK}_{SU_i}, \mathtt{msg}'_j)$;
    $\mathtt{ReplyChallenge}(\delta_j^e, SU_i)$;
  else
    Send compromise report to TA;
procedure $SU_i^2(\delta_j^e)$
  $T' \leftarrow \mathtt{TimeStamp}()$;
  if $(\mathtt{VerMac}(\mathtt{SK}_{SU_i}; N^x \parallel O^x \parallel L^x, \delta_j^e)$ & $|T' - T| \leq \Delta)$ **then**
    Send integrity certificate $C_{(x,i)}$ to SU_j^x;
  else
    Send compromise report of SU_j^x to TA;
procedure $SU_j^{x2}()$
  if (Integrity certificates were received from all neighbors within a time window) **then**
    $\mathtt{Int}_{SU_j^x} \leftarrow 1$;
  else
    $\mathtt{Int}_{SU_j^x} \leftarrow 0$;

by SU_i executes $SU_i^1()$ as follows. $SU_i^1()$ starts by initialing the timer of SU_i to start counting till next time acting as verifier. Then (in Step 2) $SU_i^1()$ records the current timestamp for future use. For each prover SU_j^x, the verifier fixes a nonce N^x (Step 5), a service S^x provided by the prover (Step 6), and a pair (I^x, o^x) of valid input-output for S^x (Step 7). The service and its input-output pairs are chosen using predefined tables of services. These tables also have the valid value of attesting memory of SU_j^x. This value is assigned to L^x in Step 8. Step 9 prepares the challenge message $\texttt{msg}_i$ which is signed using the private key of the verifier in Step 10. In Step 11, SU_i challenges SU_j^x using $\texttt{msg}_i$ and its signature.

When the prover receives the verifier challenge, it executes the method $SU_j^{x1}(\texttt{msg}_i, \delta_i)$ which verifies (in Step 13) the received message and runs the service S^x on the input I^x (in Step 15). The prover also attests its memory and prepares a reply message in Steps 16 and 17, respectively. This message is then encrypted (with the public key of the verifier) in Step 18. The encrypted message is then forwarded back to the verifier in Step 19.

Upon receiving a reply to its challenge, the versifier executes the method $SU_i^2(\delta_j^e)$ which checks the encrypted message (Step 24) using valid values it already has and the time delay since the challenge. If the result is OK, the verifier sends an integrity certificate to the prover. When the prover receives integrity certificates from all its neighbors within a certain time window, the prover is marked as genuine (Step 30) while executing the method $SU_j^{x2}()$.

2) Flying Objects Attestation (Type 2): The type 2 attestation focusing on attesting drones or Balloons is detailed in this section. Besides software integrity, this attestation type attests to hardware integrity as well. Every one of FOs moves frequently between a pair of RSUs. Provided that the elements of the pair have integrity certificates produced by type 1 attestation, they will play a critical role in the current attestation process. Suppose that the protocol attests one of the FOs, fo, that moves frequently between $\texttt{RSU}_1$ and $\texttt{RSU}_2$. In this case, fo is the prover, and $\texttt{RSU}_1$ and $\texttt{RSU}_2$ are verifiers for fo.

Algorithm 2 presents the details of type 2 attestation. The attestation process starts by $\texttt{RSU}_1$ executing the method $\texttt{RSU}_1^1()$. This method records the current timestamp (Step 1) and updates the attestation trigger counter of $\texttt{RSU}_1$, which triggers the attestation process (Step 2). $\texttt{RSU}_1$ then fixes a nonce N (Step 3), a hardware component, H, of fo (Step 5), and a pair (T, R) of a valid hardware test and its result for H (Step 6). The hardware and its test-result pairs are chosen using predefined tables of the testable hardware of fo. These tables also have the valid value of attesting memory of fo. The semantics of the rest of the algorithm are similar to that given to Algorithm 1 in the previous section.

The roles of $\texttt{RSU}_1$ and $\texttt{RSU}_2$ get swapped periodically via adjusting their clock intervals for attestation initialization.

3) Vehicle Attestation (Type 3): The vehicle attestation, relying on the previous types of attestation, is presented below in detail. We assume that the RSUs, drones, and balloons are connected to a server that has good configurations of all vehicle types on the road (all registered vehicles). To optimize communications between these entities and the configuration server, we assume that each of these entities has cache memory that contains configurations of recently attested vehicle types. Also, we assume that each of these entities, when necessary, checks the content of the cache of neighboring entities. More specifically, for example, when an RSU needs the genuine configuration of a vehicle type, the RSU can check the cache of neighboring RSUs before communicating with remote servers. This phase uses an attestation ID Aid which is initialized at the system deployment to 0.

The vehicle attestation process is triggered periodically by the TA by executing the method $\texttt{TA}^1()$ of Algorithm 3. $\texttt{TA}^1()$ starts by selecting a set of secure agents $\mathcal{A}$ that includes

Algorithm 2 Flying Objects (Drones and Balloons) Attestation.

1: **procedure** $\texttt{RSU}_1^1()$
2: $Ti \leftarrow \texttt{TimeStamp}()$;
3: Update the attestation trigger counter of $\texttt{RSU}_1$;
4: $N \leftarrow \{0,1\}^n$;
5: $H \leftarrow$ Fix a hardware component to attest in FO;
6: $(T, R) \leftarrow$ Fix a hardware test for FO and its valid result;
7: $L \leftarrow$ Valid memory attestation of FO;
8: $\texttt{msg}_1 \leftarrow N \parallel H \parallel T$;
9: $\delta \leftarrow \texttt{sig}(\texttt{SK}_{\texttt{RSU}_1}, \texttt{msg})$;
10: $\texttt{msg}_2 \leftarrow N \parallel L \parallel R \parallel Ti$;
11: $\delta_1^e \leftarrow \texttt{Enc}(\texttt{PK}_{\texttt{RSU}_2}, \texttt{msg}_2)$;
12: $\texttt{Challenge}(\texttt{msg}_1, \delta, \texttt{FO})$;
13: $\texttt{SND}(\delta_1^e, \texttt{RSU}_2)$;
14: **procedure** $\texttt{FO}^1(\texttt{MSG}_1, \delta)$
15: **if** $(\texttt{VerSig}(\texttt{PK}_{\texttt{RSU}_1}; \texttt{msg}_1, \delta))$ **then**
16: $N \parallel H \parallel T \leftarrow \texttt{msg}_1$;
17: $R' \leftarrow$ Run the test T on the hardware H during the time interval till reaching $\texttt{RSU}_2$;
18: $L' \leftarrow \texttt{MemoryAttest}(\texttt{FO})$;
19: $\texttt{msg}' \leftarrow N \parallel L' \parallel R'$;
20: $\delta_2^e \leftarrow \texttt{Enc}(\texttt{PK}_{\texttt{RSU}_2}, \texttt{msg}')$;
21: $\texttt{ReplyChallenge}(\delta_2^e, \texttt{RSU}_2)$;
22: **else**
23: Send compromise report to TA;
24: **procedure** $\texttt{RSU}_2^1(\delta_2^e, \delta_1^e)$
25: $Ti' \leftarrow \texttt{TimeStamp}()$;
26: **if** $(\texttt{Dec}(\texttt{SK}_{\texttt{RSU}_2}, \delta_2^e) == \texttt{Dec}(\texttt{SK}_{\texttt{RSU}_2}, \delta_1^e)$ & $|Ti' - Ti(\delta_1^e)| \leq \Delta)$ **then**
27: Send integrity certificate C to FO;
28: **else**
29: Send compromise report of FO to TA;
30: **procedure** $\texttt{FO}^2()$
31: **if** (Integrity certificates was received from $\texttt{RSU}_2$) **then**
32: $\texttt{Int}_{\texttt{FO}} \leftarrow 1$;
33: **else**
34: $\texttt{Int}_{\texttt{FO}} \leftarrow 0$;

Algorithm 3 Vehicle Attestation - Agent Role.

1: **procedure** TA^1()
2: $\mathcal{A} \leftarrow$ Select a set of secure SUs and FOs;
3: Update the attestation trigger counter of TA;
4: Aid $\leftarrow$ Aid $+ 1$;
5: $T_1 \leftarrow$ TimeStamp();
6: $msg_1 \leftarrow$ Aid $\parallel T_1$;
7: $\delta \leftarrow sig(SK_{TA}, msg_1)$;
8: Challenge-Agents($\mathcal{A}, msg_1, \delta$);
9: **procedure** AGENT^1(msg_1, δ)
10: $T_2 \leftarrow$ TimeStamp();
11: Aid $\parallel T_1 \leftarrow msg_1$;
12: **if** (VerSig($PK_{TA}; msg_1, \delta$) & $|T_2 - T_1| \leq \Delta$) **then**
13: $\mathcal{I}_A \leftarrow$ ID of Agent;
14: $msg_2 \leftarrow$ Aid $\parallel T_2 \parallel \mathcal{I}_A$;
15: $\delta_1^e \leftarrow Sig(SK_A, msg_2)$;
16: Challenge-Vehicles(msg_2, δ_1^e);
17: **else**
18: Report to TA;

SUs and FOs to participate in the attestation process. The choice is based on the results of types 1 and 2 attestation and on balancing the overhead to these agents by switching among them (Step 1). TA then updates its attestation trigger counter, which triggers the attestation process. Then in step 4 an attestation ID is created by incriminating Aid in Step 4. A challenge message is then prepared, signed, and sent to all agents ($\mathcal{A}$) in steps $5-8$.

When an agent receives a challenge message, it executes the method Agent^1(msg_1, δ). The agent records the current timestamp (Step 10). Then the agent checks if the received message is genuine and if it was received with an acceptable time delay (Step 12). A challenge message is then prepared and encrypted by the agent in steps 14 and 15, respectively. Finally, the agent challenges the vehicles in its range in step 16.

When a vehicle v_1 receives a challenge from an agent, it executes VEH^1(msg_2, δ_1^e) of Algorithm 4. The method starts by checking the originality of the received message (Step 4). This step also checks whether the challenge was received with acceptable delay ($|T_3 - T_2| \leq \Delta$). The value of Δ is typically a function in the size of the IoV [27]. However, a time ranging between 200 milliseconds and a few seconds would be considered acceptable. Then, v_1 checks whether the challenge message was not received before via comparing the received attestation ID (Aid) with the previously received one (CurrAid) in Step 6. Only if the results of these two tests are positive, the method continues as follows. In step 7, v_1 updates CurrAid with the new value. Then v_1 prepares a message msg_3 that includes its attestation result in steps $8-13$. This message has the current timestamp, the current attestation ID, and a singleton list of the triple $(\mathcal{I}_V, L, G)$. The elements of the triple are the ID of v_1 (denoted by $\mathcal{I}_V$), the result of attesting the memory of v_1 (denoted by L), and the geographical location of v_1 (denoted by G). The vehicle v_1 then encrypts the message msg_3 and uses it to challenge its neighboring vehicles in Step 14. We note that all along the process of vehicle attestation, encrypted forms of results are accumulated for further check by TA. Therefore, every vehicle has a list (called an attestation list) of collected vehicle attestations.

When a vehicle v_2 receives a challenge from neighboring vehicle v_1, v_2 executes the method VEH^2(msg_3, δ_2^e) of Algorithm 4. The vehicle v_2 carries a test similar to the first

Algorithm 4 Vehicle Attestation - Vehicle-to-Vehicle Communication.

1: **procedure** VEH^1(msg_2, δ_1^e)
2: Aid $\parallel T_2 \parallel \mathcal{I}_A \leftarrow msg_2$;
3: $T_3 \leftarrow$ TimeStamp();
4: **if** ($\neg$(VerSig($PK_{\mathcal{I}_A}; msg_2, \delta_1^e$) & $|T_3 - T_2| \leq \Delta$) **then**
5: Terminate procedure and report to TA;
6: **if** (CurrAid $<$ Aid) **then**
7: CurrAid $\leftarrow$ Aid;
8: $\mathcal{I}_V \leftarrow$ ID of VEH;
9: $L \leftarrow$ MemoryAttest(VEH);
10: $G \leftarrow$ GeoLocation(VEH);
11: $\mathcal{L} \leftarrow [(\mathcal{I}_V, L, G)]$
12: $msg_3 \leftarrow$ Aid $\parallel T_3 \parallel \mathcal{L}$;
13: $\delta_2^e \leftarrow Sig(SK_{\mathcal{I}_V}, msg_3)$;
14: Challenge-Vehicles(msg_3, δ_2^e);
15: **procedure** VEH^2(msg_3, δ_2^e)
16: Aid $\parallel T_3 \parallel \mathcal{L} \leftarrow msg_3$;
17: $[(\mathcal{I}_{V'}, \_, \_)] \leftarrow$ Head($\mathcal{L}$);
18: $T_4 \leftarrow$ TimeStamp();
19: **if** ($\neg$(VerSig($PK_{\mathcal{I}_{V'}}; msg_3, \delta_2^e$) & $|T_4 - T_3| \leq \Delta$)) **then**
20: Terminate procedure and report to TA;
21: **if** (CurrAid $<$ Aid) **then**
22: CurrAid $\leftarrow$ Aid;
23: $\mathcal{I}_V \leftarrow$ ID of VEH;
24: $L \leftarrow$ MemoryAttest(VEH);
25: $G \leftarrow$ GeoLocation(VEH);
26: Curr$\mathcal{L} \leftarrow [(\mathcal{I}_V, L, G)] + \mathcal{L}$;
27: Send confirmation of receiving $\mathcal{L}$;
28: **else**
29: **if** (CurrAid == Aid & Size(Curr$\mathcal{L}$ + $\mathcal{L}$) $\leq \Sigma$) **then**
30: Curr$\mathcal{L} \leftarrow$ Curr$\mathcal{L} + [\mathcal{L}]$;
31: Send confirmation of receiving $\mathcal{L}$;
32: $msg_4 \leftarrow$ Aid $\parallel T_4 \parallel$ Curr$\mathcal{L}$;
33: $\delta_3^e \leftarrow Sig(SK_V, msg_4)$;
34: **if** (Model-Contradiction(Curr$\mathcal{L}$) $\vee$ Size(Curr$\mathcal{L}$) == Σ) **then**
35: Report (msg_4, δ_3^e) to nearest agent;
36: $Q \leftarrow []$;
37: Report $\leftarrow 1$;
38: **else**
39: Challenge-Vehicles(msg_4, δ_3^e);

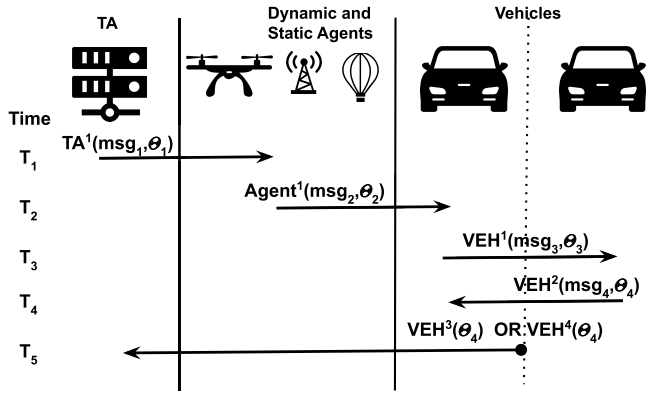

Fig. 2. Order and timeline of methods execution in type 3 (vehicle) attestation.

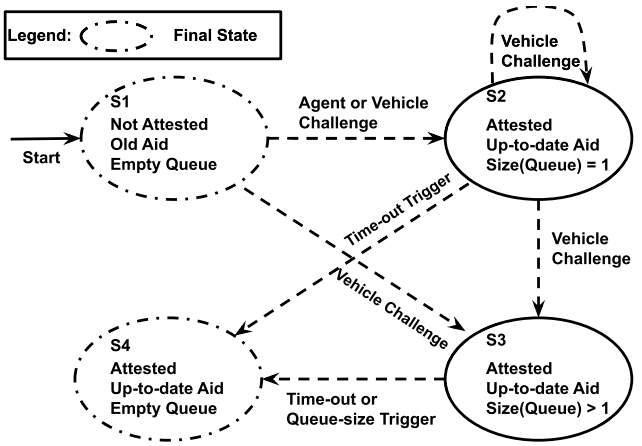

Fig. 3. Finite state machine of vehicle attestation process.

one of method $\texttt{VEH}^1(\texttt{msg}_2, \delta_1^e)$. If the result of this test is positive v_2 behaves similarly to the method $\texttt{VEH}^1(\texttt{msg}_2, \delta_1^e)$ of v_1 and does two extra steps. The first of the two steps assigns the v_2's attestation results plus the received attestation list ($\mathcal{L}$) from v_1 to its attestation list (Step 26). The second step sends a confirmation to v_1 confirming the reception of v_1's attestation list $\mathcal{L}$ (Step 27). This confirmation is necessary to avoid repeated reporting of $\mathcal{L}$ to TA. If the second test (Step 21) is not satisfied, it means that v_2 has previously received the current challenge. In this case, v_2 checks whether it can host the received attestation list (condition $\texttt{Size}(\texttt{Curr}\mathcal{L}+[\mathcal{L}]) \leq \Sigma$) of Step 29). This is the case v_2 executes only the extra pair of steps (Steps 30 and 31). We note that in Step 30, $\mathcal{L}$ is marked with extra squared parenthesis to make that at this point in the list one vehicle delivered its list of attestations to another vehicle. This is expressed formally later in Definition 1. Then v_2 prepares a new challenging message $\texttt{msg}_4$ (Step 32) and encrypts it (Step 33). Finally, in steps $34-39$, v_2 reports this message to the nearest agent in two cases. The first case is that $\texttt{msg}_4$ has two vehicles that have the same model and different attestation results ($\texttt{Model-Contradiction}(\texttt{Curr}\mathcal{L})$ in Step 34). In this case, v_2 carries an immediate reporting, because this could be a strong sign of compromised vehicle software. The second case is that the attestation list of v_2 is full (expressed as $\texttt{Size}(\texttt{Curr}\mathcal{L}) == \Sigma$) in Step 34). If the test of Step 34 is not satisfied v_2 continues with challenging neighboring vehicles.

Fig 2 concludes the main steps and methods of vehicle attestation (type 3). The figure also presents the timeline for executing the methods. Fig 3 presents a finite state machine that expresses all possible states that vehicles can go through while executing methods of Algorithm 4 during the attestation process. We later link this figure to our operational semantics of the protocol.

Algorithm 5 presents two trigger methods that every vehicle has: $\texttt{VEH}^3(\texttt{TimeStamp}() - T_{\texttt{Attest}})$ and $\texttt{VEH}^4(\texttt{List Reception})$. The method $\texttt{VEH}^3(\texttt{TimeStamp}() - T_{\texttt{Attest}})$ triggers the vehicle to report its attestation results after a predefined time has passed since attestation start. Hence the condition that triggers the method is the value of $\texttt{TimeStamp}() - T_{\texttt{Attest}}$. Therefore $T_{\texttt{Attest}}$ is the time when the vehicle has completed its attestation:

Algorithm 5 Vehicle Attestation – Vehicle Report.

1: **procedure** $\texttt{VEH}^3(\texttt{TimeStamp}() - T_{\texttt{Attest}})$
2: Report δ^e to the nearest agent;
3: $\mathcal{L} \leftarrow []$;
4: $\texttt{Report} \leftarrow 1$;
5: **procedure** $\texttt{VEH}^4(\texttt{List Reception})$
6: **if** (No confirmation was sent recently to the sender of received confirmation) **then**
7: $\mathcal{L} \leftarrow []$;
8: $\texttt{Report} \leftarrow 1$;

$T_{\texttt{Attest}}$ is T_3 (T_4) if the vehicle has executed $\texttt{VEH}^1$ ($\texttt{VEH}^2$). The method $\texttt{VEH}^4(\texttt{Confirming List Reception})$ triggers a vehicle v to empty its attestation list and marks its attestation data as reported ($\texttt{Report} \leftarrow 1$) after delivering the list to a neighboring vehicle and receives a confirmation of the list reception. To avoid any loss of attestation data, v removes the list only if it has not recently sent any list confirmation messages (Step 6 of the method).

Up on receiving attestation results, TA executes the method $\texttt{TA}^2(\delta^e)$ of Algorithm 6 which starts by extracting values of the message $\texttt{msg}_4$ in Step 2. Step 3 of the method extracts the head of the attestation list $\mathcal{L}$. This head is the attestation result of the last attested vehicle (in $\mathcal{L}$) according to the way of building $\mathcal{L}$. The tail of the list is also extracted in Step 5 which is followed by a step that records the current timestamp. Step 7 checks the originality of the received message and the time delay associated with its reception. If the test result is OK, TA processes the attestation data in $\mathcal{L}$ via executing while loop in the Steps $(9-13)$. The loop keeps extracting (Step 13) the first element of the remaining list ($\mathcal{L}'$) and stops when the list is empty (Step 9). The content in $\mathcal{L}$ is utilized using the following methods. The method $\texttt{GeoLocsExtract}$ builds a temporal map (tree) sL for the geographical locations where the attestation of vehicles has happened. Fig 4 presents an example of such maps. This map is useful for temporal and geographical forensic investigations (tracing locations of adversaries). TA then executes the method $\texttt{CompVehs}(\mathcal{L}, sL)$

Algorithm 6 `Vehicle Attestation - TA Role.`

1: **procedure** $\texttt{TA}^2(\texttt{MSG}_4, \delta_3^e)$
2: $\texttt{Aid} \parallel T_4 \parallel \mathcal{L} \leftarrow \texttt{msg}_4$;
3: $[\mathcal{I}_{V'}, L, G] \leftarrow \texttt{HEAD}(\mathcal{L})$;
4: $\mathcal{L}' \leftarrow \texttt{TAIL}(\mathcal{L})$;
5: $\texttt{CheckAID}\ (Aid)$;
6: $T_5 \leftarrow \texttt{TimeStamp}()$;
7: **if** $(\neg(\texttt{VerSig}(\texttt{PK}_{\mathcal{I}_{V'}}; \texttt{msg}_4, \delta_3^e)\ \ \&\ \ |T_5 - T_4| \leq \Delta))$ **then**
8: Take proper action;
9: **while** $\mathcal{L}' \neq []$ **do**
10: $sL \leftarrow \texttt{GeoLocsExtract}(\mathcal{I}_{V'}, G)$;
11: $s\mathcal{I}_v \leftarrow \texttt{CompVehs}(\mathcal{I}_{V'}, L)$;
12: $\mathcal{L}' \leftarrow \texttt{TAIL}(\mathcal{L}')$;
13: $[\mathcal{I}_{V'}, L, G] \leftarrow \texttt{HEAD}(\mathcal{L}')$;
14: $n\mathcal{A} \leftarrow \texttt{NextChaAgents}(sL, s\mathcal{I}_v)$;
15: $\texttt{Healing}(s\mathcal{I}_v)$;

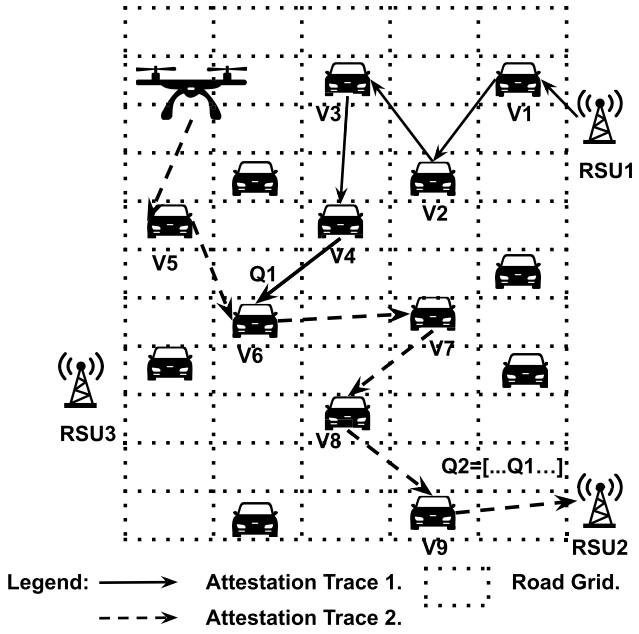

Fig. 4. Example of applying LHASIoV for geographical forensic investigations.

that checks the memory attestation results included in $\mathcal{L}$ to prepare a list $s\mathcal{I}_v$ of pairs of a compromised vehicle and its location at the attestation time. In Step 14, TA then executes the method $\texttt{NextChaAgents}(sL, s\mathcal{I}_v)$ to specify the set $n\mathcal{A}$ of agents to be challenged in the next vehicle attestation process. This set is affected by the results of the current attestation as clear from its inputs. Finally in Step 15, $s\mathcal{I}_v$ is forwarded to the healing algorithm via executing the method $\texttt{Healing}(s\mathcal{I}_v)$. In this case, HEAD and TAIL are revised versions of the ones typically used with lists. This is so as our attestation list may include members that are lists of attestation triples of vehicles i.e. the recursive definition of $\mathcal{L}$ (Definition 1 below) clarifies sheds more light on this.

The healing process is achieved as follows. From the location, speed, and direction of the compromised vehicle, TA specifies a geographical range for the current location of a compromised vehicle. Then TA sends the ID and correct attestation value of the compromised vehicle to agents in the geographical range. Once an agent finds the compromised vehicle, the agent collaborates with its neighbors to trace the vehicle and find a healing vehicle, a vehicle whose model is identical to that of the compromised one but is not compromised. Then agents copy valid software from the healing vehicle to the compromised one. If within a reasonable time window, no healing vehicle was found, then TA sends a valid copy of the software to participating agents. If the size of the software is not convenient for healing the compromised vehicle in one go, the software is sliced and consecutive agents apply consecutive software slices.

The architecture of our proposed protocol supports revealing compromised SUs and FOs in the early stage and before embarking on vehicle attestation. This has the advantage of healing compromised units with minimum delay and avoiding using compromised SUs and FOs in Type 3 attestation. Consequently, this guarantees a safe vehicular-network attestation environment.

V. Operational and Security Analysis

In this section, we present novel operational semantics to the main APIs of LHASIoV and carry out a detailed security analysis to LHASIoV. The operational semantics is useful in reasoning about our proposed protocol and is linked to its security analysis. The semantics and analysis are presented in Sections V-A and V-B, respectively. At the end of the section, we establish an interesting link between the semantics and the analysis.

A. Operational Analysis

The operational semantics of this section use the following recursive formal definition of the set of lists of vehicle attestation results built gradually in LHASIoV.

Definition 1: The set of attestation lists, $\mathcal{AL} \ni \mathcal{L}$, expressing results of LHASIoV, is defined recursively as follows:

1) $T_1 = \{(\mathcal{I}_v, T, G) \mid \mathcal{I}_v$ *is a vehicle ID,* T *is a memory attestation value,* G *is a geographical location}.*
2) $T_2 = \{[e_1, \ldots, e_n] \mid e_i \in T_1\}$.
3) $T_2 \subseteq \mathcal{AL}$.
4) $\mathcal{L}_1, \mathcal{L}_2 \in \mathcal{AL} \Longrightarrow \mathcal{L}_1 \parallel [\mathcal{L}_2] \in \mathcal{AL}$.

We note that the structure of the elements of $\mathcal{AL}$ is a bit involved: they are lists of "attestation values and lists of attestation values". This is reflected in Definition 1.4 and is in line with Step 30 of Algorithm 4.

There are two main components for our operational semantics: semantic states and semantic specifications or inference rules. The specifications guide the transit from one state to another state according to the content of LHASIoV. Definition 2 gradually builds the semantic states.

Definition 2: 1) *The set of basic states is defined as* $\mathcal{S} = \{\perp, \texttt{Compromised}, \texttt{Secure}, \texttt{RunAttest}\}$. *Every element of this set is denoted by* ω.
2) *The set of states of flying objects is defined as* $\mathcal{S}^f = \mathcal{S} \cup \{\texttt{Veh-Agent}\}$.

3) *The set of states of side objects is defined as* $\mathcal{S}^s = \mathcal{S}^d \cup \{\texttt{Drone-Agent}\}$.
4) *The set of states of vehicles is defined as* $\mathcal{S}^v = \{\texttt{Compromised}, \texttt{Genuine}, \perp\}$.
5) *The specifications of flying objects is the set of maps from IDs of flying objects to* $\mathcal{S}^f$: $m^f \in \mathcal{M}^f = \mathcal{I}(FOs) \longrightarrow \mathcal{S}^f$.
6) *The specifications of side units is the set of maps from IDs of side units to* $\mathcal{S}^s : m^s \in \mathcal{M}^s = \mathcal{I}(SUs) \longrightarrow \mathcal{S}^s$.
7) *The specifications of vehicles is the set of maps from IDs of vehicles to* $\mathcal{S}^v : m^v \in \mathcal{M}^v = \mathcal{I}(\texttt{Vehs}) \longrightarrow \mathcal{S}^v$.
8) *The set of semantic states,* $\mathcal{S}em$*, of* LHASIoV *is defined as* $\{(m^s, m^f, m^v) \mid m^s \in \mathcal{M}^s, m^f \in \mathcal{M}^f, m^v \in \mathcal{M}^v\}$. *Every element of this set is denoted by* Ω.

The initial system state is expressed as $\{(\mathcal{I}(FOs) \longrightarrow \{\perp\}, \mathcal{I}(SUs) \longrightarrow \{\perp\}, \mathcal{I}(\texttt{Vehs}) \longrightarrow \{\perp\})\}$. The states of the finite state machine presented in Fig 3 can be realized as abstractions of elements in $\mathcal{S}em$.

The following definition presents the building blocks of running LHASIoV.

Definition 3: The set of API of LHASIoV *is the set of all procedures of all algorithms constituting* LHASIoV*. A* LHASIoV *trace is a finite sequence of* LHASIoV *APIs.*

Tables II and III introduce the inference rules of our LHASIoV operational semantics. Every rule has the pattern specified in Rule 3. The pre-conditions are the conditions necessary for the network system to transfer from the state Ω_{old} to the state Ω_{new}.

$$\frac{\texttt{Pre-conditions}}{(\mathcal{L}^p, \mathcal{I}_E) \models \texttt{meth} : \Omega_{old} \longrightarrow (\Omega_{new}, \omega)} \tag{3}$$

This state transformation results from executing the method meth (LHASIoV API) by the network entity whose ID is $\mathcal{I}_E$. $\mathcal{L}^p$ denotes the set of lists of vehicle attestation values collected so far by LHASIoV: hence $\mathcal{L}^p \in \mathcal{AL}$ (Definition 1). ω denotes the overall system state from the point of view of the device executing meth. For example, Rule 4 reads as follows. Executing the API $SU_i^1()$ must start from a state (m^s, m^f, m^v) whose entity m^s specifics the state of the side unit $\mathcal{I}_{SU}$ as $\perp$ or RunAttest. In this case, the execution ends in a Secure state $(m^{s\prime}, m^{f\prime}, m^{v\prime})$ whose entity $m^{s\prime}$ specifics the state of $\mathcal{I}_{SU}$ and its neighbours as RunAttest.

TABLE II
INFERENCE RULES OF LHASIoV SEMANTICS

$$\frac{\Omega_{new} = (m^{s\prime}, m^{f\prime}, m^{v\prime}) \quad m^s(\mathcal{I}_{SU}) \in \{\perp, \texttt{RunAttest}\} \quad m^{s\prime} = m^s[\mathcal{I}_{SU} \mapsto \texttt{RunAttest}, \mathcal{I}^i_{SU} \mapsto \texttt{RunAttest}, \forall \mathcal{I}^i_{SU} \textit{ neighbor of } \mathcal{I}_{SU}]}{([], \mathcal{I}_{SU}) \models SU_i^1() : (m^s, m^f, m^v) \rightarrow (\Omega_{new}, \texttt{Secure})} \tag{4}$$

$$\frac{\Omega_{new} = (m^{s\prime}, m^{f\prime}, m^{v\prime}) \quad m^s(\mathcal{I}_{SU}) = \texttt{RunAttest} \quad \omega = (\texttt{msg}_i \textit{ agrees with } \delta_i)\ \texttt{Secure} : \texttt{Compromised}}{([], \mathcal{I}_{SU}) \models SU_j^{x1}(msg_i, \delta_i) : (m^s, m^f, m^v) \rightarrow (\Omega_{new}, \omega)} \tag{5}$$

$$\frac{\omega = (\texttt{VerMac}(\texttt{SK}_{SU_i}; N^x \parallel O^x \parallel L^x, \delta^e_j) \ \&\ |T' - T| \leq \Delta)\ \texttt{Secure} : \texttt{Compromised}}{([], \mathcal{I}_{SU}) \models SU_i^2(\delta^e_j) : (m^s, m^f, m^v) \rightarrow (\Omega_{new}, \omega)} \tag{6}$$

$$\frac{\Omega_{new} = (m^{s\prime}, m^{f\prime}, m^{v\prime}) \quad m^s(\mathcal{I}_{SU}) = \texttt{RunAttest} \quad \omega = (\textit{All neighbors replied})\ \texttt{Secure} : \texttt{Compromised} \quad m^{s\prime} = m^s[\mathcal{I}_{SU} \mapsto \omega]}{([], \mathcal{I}_{SU}) \models SU_j^{x2}() : (m^s, m^f, m^v) \rightarrow (\Omega_{new}, \omega)} \tag{7}$$

$$\frac{\Omega_{new} = (m^{s\prime}, m^{f\prime}, m^{v\prime}) \quad \mathcal{I}'_{\texttt{RSU}} \textit{ is corresponding RSU} \quad m^s(\mathcal{I}_{\texttt{RSU}}) = m^s(\mathcal{I}'_{\texttt{RSU}}) = \texttt{Secure} \quad m^{s\prime} = m^s[\mathcal{I}_{FO} \mapsto \texttt{RunAttest}, \forall \mathcal{I}_{FO} \textit{ assigned to } \mathcal{I}_{\texttt{RSU}}]}{([], \mathcal{I}_{\texttt{RSU}}) \models \texttt{RSU}_1^1() : (m^s, m^f, m^v) \rightarrow (\Omega_{new}, \texttt{Secure})} \tag{8}$$

$$\frac{\Omega_{new} = (m^{s\prime}, m^{f\prime}, m^{v\prime}) \quad m^s(\mathcal{I}_{FO}) = \texttt{RunAttest} \quad \omega = (\texttt{msg}_1 \textit{ agrees with } \delta)\ \texttt{Secure} : \texttt{Compromised}}{([], \mathcal{I}_{FO}) \models \texttt{FO}^1(\texttt{msg}_1, \delta) : (m^s, m^f, m^v) \rightarrow (\Omega_{new}, \omega)} \tag{9}$$

$$\frac{m^s(\mathcal{I}_{\texttt{RSU}}) = \texttt{Secure} \quad \omega = (\texttt{Dec}(\texttt{SK}_{\texttt{RSU}_2}, \delta^e_2) == \texttt{Dec}(\texttt{SK}_{\texttt{RSU}_2}, \delta^e_1) \ \&\ |Ti' - Ti(\delta^e_1)| \leq \Delta)\ \texttt{Secure} : \texttt{Compromised}}{([], \mathcal{I}_{\texttt{RSU}}) \models \texttt{RSU}_2^1(\delta^e_2, \delta^e_1) : (m^s, m^f, m^v) \rightarrow (\Omega_{new}, \omega)} \tag{10}$$

$$\frac{\Omega_{new} = (m^{s\prime}, m^{f\prime}, m^{v\prime}) \quad m^s(\mathcal{I}_{FO}) = \texttt{RunAttest} \quad \omega = (\texttt{RSU}_2 \textit{ replied})\ \texttt{Secure} : \texttt{Compromised} \quad m^{s\prime} = m^s[\mathcal{I}_{FO} \mapsto \omega]}{([], \mathcal{I}_{FO}) \models \texttt{FO}^2() : (m^s, m^f, m^v) \rightarrow (\Omega_{new}, \omega)} \tag{11}$$

B. Security Analysis

In this section, we prove the security of LHASIoV, i.e. LHASIoV guarantees that adversaries (Adv_s) are not able to report a genuine system state (genuine attest) to the TA for software-compromised systems. In cases where adversaries (Adv_h) can launch physical attacks, LHASIoV guarantees that they can not fake valid system states for physically genuine devices. This is essential to restrict physical-attack scalability and to make such attacks not cost-efficient for larger-scale networks. This section also proves that LHASIoV is resilient to many other security attacks.

The following definition formulates a common concept, adversarial experiment, in literature [14], [28], [29] to define the security of a network attestation protocol p. In the context of this concept, an adversary can trigger TA to start an attestation process to watch and attack the network system.

Definition 4: An adversarial experiment is a map

$$\mathcal{SecAtt}(adv, n, p, A_n, Mac_p, Sig_p) = [o_1, \ldots, o_n], \tag{19}$$

such that:

- *n is the number of initialized network entities that the adversary, adv, has access to and that run the attestation scheme, p. The communication among the adversary and network entities follows our threat model presented earlier in the paper.*
- *A_n is the number of times in which adv can communicate with network entities. This number is polynomial because adv is computationally bound.*
- *Mac_p and Sig_p are security parameters depending on mac and signature schemes, respectively, used in p.*
- *For each $1 \leq i \leq n, o_i \in \{0, 1\}$. The vector $[o_1, \ldots, o_n]$ denotes the output returned after performing the adversarial experiment. Hence assigning a value 1 for an entry o_i denotes a secure state for the network entity E_i.*

We now present Theorems 1 and 2 to analyze the security of LHASIoV against software and hardware adversaries respectively. The proof of the theorem below is similar for different

TABLE III
INFERENCE RULES OF LHASIoV SEMANTICS

$$\frac{\Omega_{new} = (m^{s\prime}, m^{f\prime}, m^{v\prime}) \qquad m^s(\mathcal{I}_a) = \texttt{Secur}, \forall \mathcal{I}_a \in \mathcal{A} \quad m^{s\prime} = m^s[\mathcal{I}_a \mapsto \texttt{Veh-Agent}, \forall \mathcal{I}_a \in \mathcal{A}]}{([], \mathcal{I}_{\texttt{TA}}) \models \texttt{TA}^1() : (m^s, m^f, m^v) \rightarrow (\Omega_{new}, \texttt{Secure})} \tag{12}$$

$$\frac{\begin{array}{c} m^s(\mathcal{I}_a) = \texttt{Veh-Agent} \\ \omega = (\texttt{VerSig}(\texttt{PK}_{\texttt{TA}}; \texttt{msg}_1, \delta) \\ \& \ |T_2 - T_1| \leq \Delta) \ \texttt{Secure} : \texttt{Compromised} \end{array}}{([], \mathcal{I}_a) \models \texttt{Agent}^1(\texttt{msg}_1, \delta) : (m^s, m^f, m^v) \rightarrow (\Omega_{new}, \omega)} \tag{13}$$

$$\frac{\begin{array}{c} \Omega_{new} = (m^{s\prime}, m^{f\prime}, m^{v\prime}) \\ \mathcal{L}^p = [(\mathcal{I}_V, L, G)] \qquad \omega = (\texttt{VerSig}(\texttt{PK}_{\mathcal{I}_A}; \texttt{msg}_2, \delta_1^e) \\ \& \ |T_3 - T_2| \leq \Delta) \ \texttt{Secure} : \texttt{Compromised} \\ m^s(\mathcal{I}_V) = \perp \qquad m^{s\prime} = m^s[\mathcal{I}_V \mapsto \texttt{Run-Attest}] \end{array}}{(\mathcal{L}^p, \mathcal{I}_V) \models \texttt{VEH}^1(\texttt{msg}_2, \delta_1^e) : (m^s, m^f, m^v) \rightarrow (\Omega_{new}, \omega)} \tag{14}$$

$$\frac{\begin{array}{c} m^s(\mathcal{I}_V) \in \{\perp, \texttt{Run-Attest}\} \quad \Omega_{new} = (m^{s\prime}, m^{f\prime}, m^{v\prime}) \\ [(\mathcal{I}_V, L, G)] \in \mathcal{L}^p \qquad \omega = (\texttt{VerSig}(\texttt{PK}_{\mathcal{I}_{V'}}; \texttt{msg}_3, \delta_2^e) \\ \& \ \neg \texttt{Model-Contradiction}(\mathcal{L}^p) \\ \& \ |T_4 - T_3| \leq \Delta) \ \texttt{Secure} : \texttt{Compromised} \\ m^{s\prime} = m^s[\mathcal{I}_V \mapsto \texttt{Run-Attest}] \end{array}}{(\mathcal{L}^p, \mathcal{I}_V) \models \texttt{VEH}^2(\texttt{msg}_3, \delta_2^e) : (m^s, m^f, m^v) \rightarrow (\Omega_{new}, \omega)} \tag{15}$$

$$\frac{\texttt{Attestation time is over}}{([], \mathcal{I}_V) \models \texttt{VEH}^3() : \Omega \rightarrow (\Omega, \texttt{Secure})} \tag{16}$$

$$\frac{\texttt{Queue Reception Confirmation}}{([], \mathcal{I}_V) \models \texttt{VEH}^4() : \Omega \rightarrow (\Omega, \texttt{Secure})} \tag{17}$$

$$\frac{\begin{array}{c} \omega = (\texttt{VerSig}(\texttt{PK}_{\mathcal{I}_{V'}}; \texttt{msg}_4, \delta_3^e) \\ \& \ \mathcal{L}^p \ \texttt{has no compromised vehicles} \\ \& \ |T_5 - T_4| \leq \Delta) \ \texttt{Secure} : \texttt{Compromised} \end{array}}{(\mathcal{L}^p, \mathcal{I}_{TA}) \models TA^2(\texttt{msg}_4, \delta_3^e) : \Omega \rightarrow (\Omega, \omega)} \tag{18}$$

system entities and hence we focus our proof on an instance of the main system entity, a vehicle v_i.

Theorem 1: For every software adversary Adv_s, LHASIoV is secure against Adv_s. In other words, there exists a negligible map $\mathcal{S}$ such that for every network entity E_i with compromised software during the attestation process: $Pro[\mathcal{SecAtt}(Adv_s, n, \texttt{LHASIoV}, A_n, Mac_p, Sig_p)(i) = 1] \leq \mathcal{N}(\mu)$, *where μ is a polynomial map in the parameters A_n, Mac_p, Sig_p, and $p =$ LHASIoV. Moreover, LHASIoV is resilient to replay and denial of service attacks launched by Adv_s.*

Proof: We prove that Adv_s is not able to fabricate a genuine system state for a vehicle v_i whose software is compromised during its attestation. To this end, we prove that Adv_s can not contribute a genuine attestation report, for v_i, that is eventually delivered to and checked by TA. Contributing such a report requires Adv_s to build a message pair $(\texttt{msg}_3, \delta_2^e)$ that includes genuine attestation values for v_i. We note that δ_2^e is authenticated with genuine key $\texttt{SK}_{\mathcal{I}_V}$. The vehicle attestation process implies sending this message to one of the RSUs directly or to another vehicle which continues distributing the message until it eventually reaches the RSUs. A genuine vehicle will only spread a received list of attestation messages if contains no contradicting values for memory attestations of vehicles of the same model. A genuine vehicle will also re-authenticate the attestation list after adding its attention messages to it. Therefore, without breaking the security of this authentication process, Adv_s can not contribute a genuine message for the under-execution attestation process. This is so as the current attestation ID Aid and timestamps contribute to the authenticated message. Bearing in mind that Adv_s can not break the security of the vehicle Trusted Execution Environment (TEE) which stores and executes the code of LHASIoV and stores Aid and the key $\texttt{SK}_{\mathcal{I}_V}$, it is evident that Adv_s can not access these values. Therefore, for the vehicle, v_i whose software is compromised during its attestation, Adv_s can not produce a genuine attestation message for the running attestation process. This is also because the memory attestation of the vehicle produces an invalid value in this case. Eventually, messages of v_i are transmitted to TA via one of the RSUs, and a vehicle compromised message gets discarded.

Similarly, it would not work for Adv_s to reply using reports stored from other genuine vehicles or past reports of V_i. This is so because the stored reports include an invalid attestation message for v_i in the current attestation process. Hence changing the vehicle messages results in invalid attestation reports, and Adv_s would need again to contribute a valid message. In other words, old vehicle messages get discarded by TA because of their invalid attestation process Aid. All in all, Adv_s can not deliver a valid attestation message for any vehicle with compromised software. Hence, Adv_s can not overcome the attestation using a compromised vehicle (during its attestation time). Therefore LHASIoV has a mechanism for prohibiting Adv_s from intentionally spreading fake reports. This mechanism is executed gradually and simultaneously on all vehicles contributing to the attestation process. Therefore, vehicles collaborate on stopping compromised messages. As a result, Adv_s can not carry a denial of service attack in the presence of LHASIoV. This is partially due to the fact that an invalid attestation value for a vehicle has a chance of being discovered at another vehicle and also invalidates the whole list of attestation values. □

Theorem 2: For every hardware adversary Adv_h, LHASIoV is secure against Adv_h. In other words, there exists a negligible map $\mathcal{H}$ such that for every network entity E_i that has a genuine TEE but a compromised software during the attestation process: $Pro[\mathcal{SecAtt}(Adv_h, n, \texttt{LHASIoV}, A_n, Mac_p, Sig_p)(i) = 1] \leq \mathcal{H}(\gamma)$, *where γ is a polynomial map in the parameters A_n, Mac_p, Sig_p, and $p =$ LHASIoV. Moreover, LHASIoV is resilient to replay attacks launched by Adv_h.*

Proof: We need to prove that Adv_h can not fake a valid system message for a vehicle v_i that has uncompromised hardware and compromised software during its attestation time. Following similar steps as in the proof of Theorem 1, What we need to prove amounts to proving that Adv_h can not contribute valid attestation messages on behalf of the vehicle. We notice that these messages are meant to eventually (via one of the RSUs) arrive and be checked by TA. Also, our assumption is that Adv_h can access TEE of a group of other neighboring vehicles, V, rather than v_i. Therefore, it is assumed that Adv_h knows the secret parameters of vehicles in

this group and can modify their LHASIoV execution phase. Hence, Adv_h is able to produce valid attestation messages for vehicles in V. Therefore these messages would pass the authentication test at the TA.

However, to produce a valid list of attestation values $\mathcal{L}$, this list also hosts a valid attestation report for v_i. We assume that TEE of v_i is genuine, and hence it is not possible to forge its attestation values in the attestation list. Additionally and similarly to proof of Theorem 1, LHASIoV resists replay attacks that could be attempted by $\mathcal{A}dv_h$. □

Theorem 3: LHASIoV can resist Man-in-The-Middle (MTM) and different impersonation attacks.

Proof: We assume an adversary Adv that listens to attestation messages exchanged between a vehicle and other system entities. These messages include $\mathtt{msg}_1, \mathtt{msg}_2, \mathtt{msg}_3$, $\mathtt{msg}_4, \delta, \delta_1^e, \delta_2^e$, and δ_3^e. The objective of Adv in this case is to create new messages that resemble the original attestation ones. Several parameters such as geographical locations and memory attestations of vehicles and IDs of system entities were used to create these messages. Such parameters could be collected by Adv via listening to genuine messages. However, Adv does not know the secret keys of system entities. Since every sent message is associated with its signature (built using the secret keys), any modification by the Adv on any message will be discovered on the message delivery. Hence it is safe to elucidate that attestation messages can not be faked by Adv without knowledge of the secret key of the sender. Therefore, LHASIoV is safe against the man-in-the-middle attack.

Similarly, as long as the device TEE is secure and the secret key is not accessible by Adv, no adversary can prepare an attestation message on behalf of system entities. Therefore, LHASIoV is secure against different impersonation attacks: vehicle-impersonation attack, RSUs-impersonation attack, ACSs-impersonation attack, drone-impersonation attack, and balloon-impersonation attack. □

Definition 5 and Theorem 4 establish the linkage between our proposed operational semantics and our security analysis.

Definition 5: A semantic state $s = (m^s, m^f, m^v) \in \mathcal{S}em$ is secure if s satisfies the inequalities of Theorems 1 and 2.

Theorem 4: Suppose that for a LHASIoV-trace tr, the judgment $(\mathcal{L}^p, \mathcal{I}_E) \models tr : \Omega_{old} \longrightarrow (\Omega_{new}, \omega)$ is produced by composing applications of relevant inference rules of our LHASIoV operational-semantics. Suppose also that Ω_{old} and the network are secure during rules application. Then it must be the case that Ω_{new} is also secure.

The proof of Theorem 4 is by structure induction on the inference rules of Tables II and III.

Our proposed protocol, LHASIoV, is resilient to the black-hole and Sybil attacks. This is so that all vehicles on the system must be registered with unique IDs. Moreover, the vehicle attestation process implies distributing attestation results to an RSU directly or to a set of vehicles in the range of the sending vehicle. Hence the effect of a compromised vehicle can be quickly revealed as the same data is delivered to different vehicles.

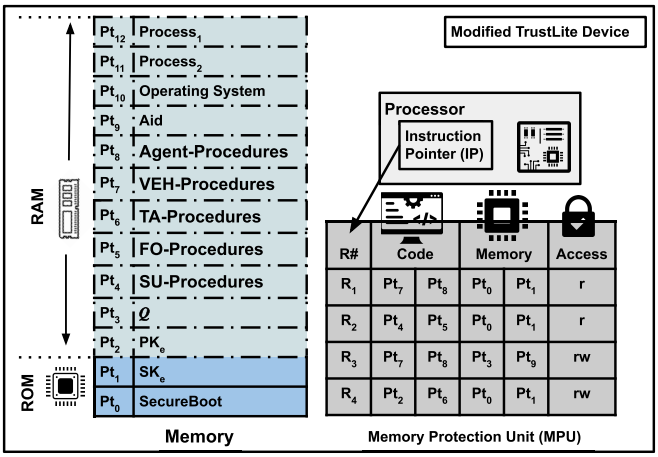

Fig. 5. Proof-of-concept implementation for LHASIoV.

VI. Implementation and Evaluation

In this section, we introduce implementation details and assess the LHASIoV performance. The tasks in this section are categorized into three categories. We first introduce a proof-of-concept implementation for LHASIoV which is based on the security architecture, TrustLite which in turn is built on a Siskiyou Peak [13] owned by *Intel*. This is done in Section VI-A. Then in Section VI-B, we present detailed and multi-facets comparisons of LHASIoV against related state-of-the-art schemes [6], [14], [15]. Finally, in Section VI-C, we prove and assess the practical feasibility of LHASIoV. This is done by implementing LHASIoV in Omnetpp, a popular networking simulation tool, associated with the Castalia simulator. We make available result files from the simulations tool that we employed to assess LHASIoV, in a public repository.[2] We carried out the experiments on a machine that has the following specifications: Dell (Vostro) Intel(R) Core(TM) i7-3612 QM CPU @ 2.10 GHz, 8.00 GB RAM on Windows 10 (64-bits) OS.

A. Prototype

This section presents a proof-of-concept implementation for our proposed protocol, LHASIoV. Our implementation is based on the security architecture, TrustLite, that is designed for embedded systems [30]. TrustLite is built on a robust (concerning confidentiality and authenticity) research system, named Siskiyou Peak [13] and built by *Intel*. From such systems and with the assistance of hardware, our implementation inherits and adopts the isolation of real-time execution (called trustlets in this case) against system entities. This isolation enables protecting critical entities against malicious access. Fig 5 presents the details of our TrustLite implementation.

Our implementation has a Memory Protection Unit (MPU) restricting the data (which is secure-boot based) access only to owner trustlets of that data. Hence, MPU is useful to restricting access to hardware entities including peripherals. As a pair of

[2]https://github.com/maelzawawy/LHASIoV

TABLE IV
COMPARING SECURITY AND FUNCTIONALITY ATTRIBUTES OF LHASIoV AGAINST THE STATE-OF-THE ART SCHEMES

#	Security\Functionality characteristics	[6]	[14]	[15]	**LHASIoV**
1	IoV-oriented scheme	✓	×	✓	✓
2	Resilience to single-point failure problems	×	×	✓	✓
3	Supporting drone assistance	×	×	×	✓
4	Supporting vehicle healing	×	×	×	✓
5	Tracing vehicle geographic locations at attestation time	×	×	×	✓
6	Simultaneously attesting system components of different types	×	×	×	✓
7	Resilience to hardware adversary	✓	✓	×	✓
8	Resilience to reply attacks	✓	✓	✓	✓
9	Resilience to man-in-the-middle attacks	✓	×	✓	✓

The mark ✓ (×) is used to express that the corresponding scheme satisfies (does not satisfy) the characteristic.

advantages of using TrustLite, rather than other similar architectures, the rules in MPU of our implementation are programmable (not static) and our implementation handles interrupting trustlets.

We implemented main pieces of LHASIoV (such as $\texttt{TA}^1()$, $\texttt{AGENT}^1()$, and $\texttt{VEH}^1()$) as trustlets (isolated processes). Moreover, we designed MPU to control accessing RAM and ROM in a manner that allows only the methods of LHASIoV to access the private data of the protocol. As an example, in Fig 5, according to the rule R_3, the attestation ID and list can only be written by the vehicle and agent methods. However, according to the rule R_1, the same methods can only read secret keys.

B. Costs and Comparisons

We compare our proposed protocol, LHASIoV, against existing related attestation schemes in the area of IoV and dynamic networks such as [6], [14], and [15]. The comparison is in terms of security and functionality characteristics, communication, computational, and storage costs.

The security and functionality characteristics that we use to compare LHASIoV against related protocols include IoV-orientation, drone assistance, formal semantic support, resilience to hardware adversary, resilience to man-in-the-middle attacks, replay attacks, and impersonation. Table IV shows the result of this comparison. The table shows that LHASIoV satisfies security and functionality requirements (essential for modern systems of drone-assisted IoV) that are not satisfied by existing protocols.

During the RSUs and ACSs attestation phase, the messages $\texttt{msg}_i, \delta_i, \delta_j^e$ (or a compromise report), an integrity certificate $C_{(x,i)}$ (or a compromise report) are communicated among registered SUs. Table V presents our assumptions about lengths (in bits) of different data pieces composing communicated messages in LHASIoV. The message $\texttt{msg}_i$ requires $160+32+32 = 224$ bits. Therefore, the total communication costs of the procedures $\texttt{SU}_i^1()$, $\texttt{SU}_j^{x1}()$, $\texttt{SU}_i^2()$ and $\texttt{SU}_j^{x2}()$ are $224 + 256$, at most 256, 32, and 0 bits, respectively. Hence, the total communication cost of RSUs and ACSs attestation phase in LHASIoV is at most $224 + 256 + 256 + 32 = 768$ bits.

TABLE V
ASSUMPTIONS FOR THE LENGTH OF DIFFERENT MESSAGE PIECES COMMUNICATED IN LHASIoV

Message piece	Length in bits
Timestamp	32
ID (for service, hardware, vehicle, or attestation)	32
Geographical location	32
Valid input (for service or hardware)	32
Valid result (from service or hardware)	32
Integrity certificate	32
Compromise report	32
Public or private key	32
Memory attestation result	32
Nonce (random number)	160
Signature	256
Cryptographic hash function	256

TABLE VI
COMMUNICATION COSTS OF PROCEDURES USED IN VEHICLE ATTESTATION PHASE OF LHASIoV

Procedure	Communication cost in bits
$\texttt{TA}^1()$	$64 + 256 = 320$
$\texttt{AGENT}^1()$	$96 + 256 = 352$
$\texttt{VEH}^1()$	$32 + 32 + (32 + 32 + 32) + 256 = 416$
$\texttt{VEH}^2()$	$\leq 320 + n * 96$ (assuming $\lvert\mathcal{L}\rvert = n$)
$\texttt{VEH}^3()$	256
$\texttt{VEH}^4()$	0
$\texttt{TA}^2()$	0

During the drones and balloons attestation phase, the messages $\texttt{msg}_1, \delta, \delta_1^e, \delta_2^e$ (or a compromise report), an integrity certificate C (or a compromise report) are communicated among registered RSUs and FOs. Using Table V, the message $\texttt{msg}_1$ requires $160 + 32 + 32 = 224$ bits. Therefore, the total communication costs of the procedures $\texttt{RSU}_1^1()$, $\texttt{FO}^1()$, $\texttt{RSU}_2^1()$ and $\texttt{FO}^2()$ are $224 + 256 + 256$, at most 256, 32, and 0 bits, respectively. Hence, the total communication cost of drones and balloons attestation phase in LHASIoV is at most $224 + 256 + 256 + 256 + 32 = 1024$ bits. Similarly, the total communication costs of the procedures, used during the vehicle's attestation phase, are presented in Table VI. Hence, the total communication cost of the vehicles attestation phase in LHASIoV is less than or equal to $320+352+416+320+n*96+256 = 1664+n*96$ bits, where n is the number of vehicle results in the list $\mathcal{L}$.

Table VII and Fig 6 present our comparative study for communication costs of vehicle attestation (Type 3 attestation - the relevant part) of LHASIoV against related protocols. The study confirms that LHASIoV requires less communication cost than the related schemes. The formulas of costs in Table VII prove that sending data through LHASIoV does not scarify other performance parameters and hence does not raise any trade-off issues. We opted for measuring costs in bits instead of using the aspect of bandwidth costs to enable smooth comparisons with state-of-the-art protocols that typically use bits. However, we did measure the performance of our proposed protocol in bandwidth consumption as Table X below shows. A connection bandwidth of 20 MHz was employed in the experiments.

To carry out a computational cost analysis, we assume that t_{se} and t_c denote the time needed to sign or encrypt a message and to calculate the concatenation of two messages, respectively. The types one, two and three of LHASIoV require $4 * t_{se} + 6 * t_c$, $6 * t_{se} + 7 * t_c$ and $8 * t_{se} + 7 * t_c$,

TABLE VII

COMPARING COMMUNICATION COST OF LHASIoV AGAINST RELATED STATE-OF-THE ART SCHEMES

Scheme	No. of messages	No. of bits
[6]	6	$m * 256$
[14]	7	$832 + m * 256$
[15]	9	$1760 + m * 288$
LHASIoV (Type 3 attestation)	5	$1664 + n * 96$

$n = |\mathcal{L}|$ & m is No. vehicles.

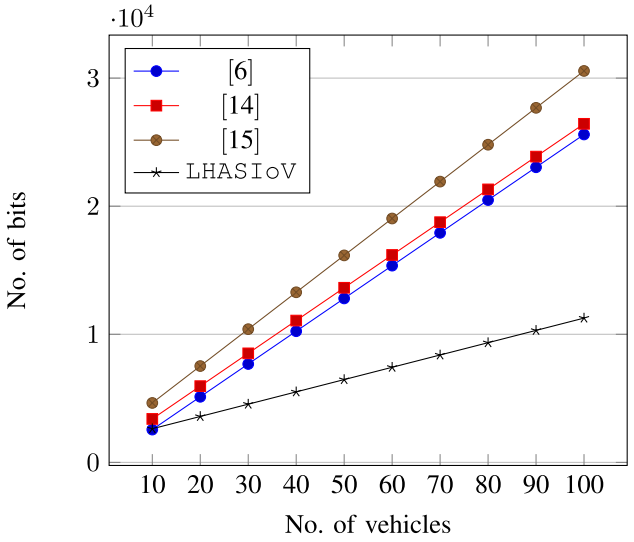

Fig. 6. Comparing Communication cost of LHASIoV against related state-of-the art schemes for different number of vehicles.

TABLE VIII

COMPARING COMPUTATIONAL COST OF LHASIoV AGAINST RELATED STATE-OF-THE ART SCHEMES

Scheme	Computational Cost
[6]	$m * (t_{se} + t_c)$
[14]	$(3 + 5 * m) * t_{se} + (6 + 8 * m) * t_c$
[15]	$14 * t_{se} + 19 * t_c$
LHASIoV	$18 * t_{se} + 20 * t_c$

m is No. vehicles.

respectively. Thus, the total computational cost in LHASIoV is $18 * t_{se} + 20 * t_c$. The efficiency of LHASIoV is partially proved by the limited number of main operations used in it. Table VIII presents our comparative study for computational costs of LHASIoV against related protocols. The study reveals that the computational costs of LHASIoV are, in most cases, less than or comparable to those of related schemes.

During the RSUs and ACSs attestation phase, every one of SUs needs to store a valid pair of input-output of a service and an integrity indicator value. Therefore this stage requires a storage cost of $3 * 16 = 48$ bits. Similarly, the drones and balloons attestation phase requires a storage cost of $3 * 16 = 48$ bits. However, the vehicles attestation phase requires storing an attestation ID (16 bits), a time stamp (32 bits), a list of attestation values for n vehicles ($n * 96$), and an integrity indicator value (16 bits). Therefore the storage cost of this phase is $48 + n * 96$ bits. Hence the total storage cost of LHASIoV is $144 + n * 96$ bits. Table IX presents our comparative study for storage costs of LHASIoV against related state-of-the-art protocols. The study shows that in realistic IoV-networks with real size, the storage costs of LHASIoV are less than those of related schemes.

TABLE IX

COMPARING STORAGE COST OF LHASIoV AGAINST RELATED STATE-OF-THE ART SCHEMES

Scheme	Storage cost (in bits)
[6]	$m * 256$
[14]	$48 + m * 112$
[15]	$m * 160$
LHASIoV	$144 + n * 96$

$n = |\mathcal{L}|$ & m is No. vehicles.

TABLE X

SIMULATION CHARACTERISTICS

Network characteristic	Value
Operating system	Windows 10
Simulation tools	Omnetpp (equipped with Castalia)
No. of TA	1
No. of FOs	2 (a drone and a balloon)
No. of ACSs	2
No. of RSUs	2
No. of experimental scenarios	10
No. of system servers	1
Range of total no. of vehicles	$50 - 200$
Range of no. of compromised vehicles	$1 - 6$
Connection bandwidth	20 MHz
Range of no. of FOs speed	$3 - 15$ mph
Range of no. of vehicles speed	$5 - 40$ mph
Routing protocol	Multi-path Rings
Noise bandwidth	194 MHz
Communication	802.15.4 MAC

All in all, our comparative study ensures that LHASIoV is more efficient and scalable than related state-of-the-art schemes. This becomes even more evident when we notice that the network system treated by LHASIoV is more general (for example supporting drone assistance) than that of most of the related work.

C. Experimental Performance

On top of the well-known networking simulation tool, Omnetpp, and its famous package, Castalia simulator, our simulation of LHASIoV is built. The main objective of our simulation is to prove the practical computational feasibility of LHASIoV. Our simulation is designed to accommodate the details of the system model of this paper presented in Section III. In a public repository,[3] we share the main result files of our simulation. Castalia APIs can be applied to these files to reproduce the results presented below. The main characteristics of our experiments are shown in Table X. Our experiments are simulated on a road that is of length 200 m and width 50 m. The road has 2 RSUs and 2 ACSs units distributed on the roadsides. Our simulation system relies on one TA unit. The system is also equipped with a drone and a balloon that moves linearly and frequently between the road ends. The road has several vehicles that move at different speeds as specified below in our experimental scenarios. The initial positioning of vehicles on the road has the shape of a matrix. The drone and balloon are initially deployed on opposite sides of the road. Moreover, to extend the realistic aspect of our experiments,

[3]https://github.com/maelzawawy/LHASIoV

TABLE XI
SCENARIOS USED FOR TESTING PRACTICAL PERSPECTIVES OF LHASIoV

ID^{ES}	V#	V^S	D^S	B^S
ES1	50	{(10,5),(20,10),(20,15)}	10	3
ES2	75	{(30,10),(30,15),(15,20)}	10	5
ES3	100	{(60,10),(40,20)}	12	3
ES4	100	{(40,15),(40,20),(20,25)}	15	5
ES5	125	{(50,10),(50,15),(25,20)}	10	5
ES6	150	{(100,15),(50,25)}	12	3
ES7	150	{(60,15),(60,20),(30,25)}	15	5
ES8	175	{(70,10),(70,15),(35,20)}	10	5
ES9	200	{(130,30),(70,40)}	12	3
ES10	200	{(80,15),(80,20),(40,25)}	15	5

ID^{ES}: Experimental Scenarios ID, V#: Total No. of vehicles, V^S: Pairs of vehicle counts and their speeds , D^S: Drone speed, B^S: Balloon speed.

TABLE XII
RESULTS OF RUNNING LHASIoV ON OUR EXPERIMENTAL SCENARIOS FOR TESTING PRACTICAL PERSPECTIVES OF THE PROTOCOL

ID^{ES}	T_A^1	T_A^2	T_A^3	A_E	P^T	P^R	$I^V\#$
ES1	2.04	2.55	5.57	6.798	40.088	46.579	5
ES2	1.25	1.56	4.15	6.798	75.634	87.231	1
ES3	1.62	2.18	5.27	6.798	101.71	118.626	2
ES4	1.80	2.06	5.27	6.798	112.972	133.206	2
ES5	2.12	2.49	6.66	6.797	132.932	158.886	5
ES6	1.27	1.76	5.93	6.798	128.248	153.618	6
ES7	0.79	1.28	5.27	6.798	119.369	150.484	4
ES8	1.51	1.77	7.28	6.797	150.846	188.763	6
ES9	1.66	2.50	7.75	6.797	154.372	217.256	2
ES10	1.48	2.02	7.39	6.797	149.005	191.652	4

ID^{ES}: Experimental Scenarios ID, T_A^1: Average time to complete attesting , SUs (sec) T_A^2: Average time to complete attesting FOs (sec), T_A^3: Average time to complete attesting vehicles (sec), A_E: Average consumption of energy per node[mJ], P^T: Average number of transmitted packets, P^R: Average number of received packets, $I^V\#$: No. Infected vehicles.

every scenario selects randomly several vehicles and assumes that their software is compromised.

Our experiments were carried out using the experimental scenarios outlined in Table XI. We present the results obtained for these scenarios in Table XII.

For each experimental scenario, as presented in Table XII, our experiments measure the average time needed to complete attesting SUs (T_A^1), the average time needed to complete attesting FOs (T_A^2), the average time needed to complete attesting vehicles (T_A^3), the average consumption of energy per node (A_E), the average number of transmitted packets (P^T), and average number of received packets (P^R), and the number of infected vehicles ($I^V\#$). Fig 7 shows the development of timings T_A^1, T_A^2, and T_A^3 as the number of vehicles increases in experimental scenarios. We notice that the times T_A^1 and T_A^2 are very close for all scenarios. This is reasonable when we notice that the side units have the same count for all scenarios and they are static entities. We also notice that the increase in T_A^3 with an increasing number of vehicles is limited which supports the practical feasibility of LHASIoV and shows its applicability on huge IoV networks. Another interesting notice is that the increase in vehicle numbers and their mobility in scenarios ES7 and $ES10$ lead to a decrease in the attestation time T_A^3, compared to scenarios ES6 and $ES9$, respectively. This is so as the chance of delivering messages increases with increasing mobility. The experiments prove a limited energy consumption of around 6.798 mJ, per node. We also note, as pictured in Fig 8, that for each scenario, the number of received packets is larger than the number of transmitted

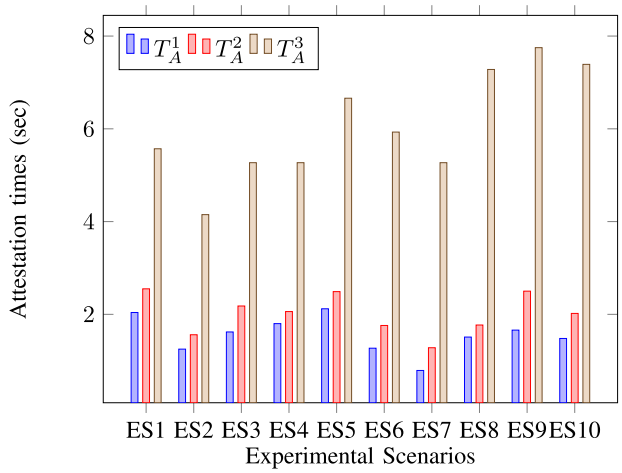

Fig. 7. Average times needed for Type 1 (T_A^1), Type 2 (T_A^2), and Type 3 (T_A^3) attention in LHASIoV.

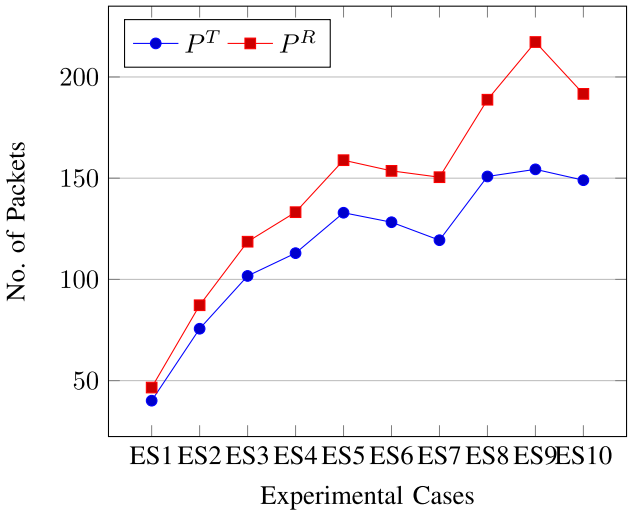

Fig. 8. Received packets verses transmitted ones for experimental scenarios of LHASIoV.

ones. This is due to the nature of LHASIoV which includes delivering the same attestation message to many vehicles. This also confirms the practical side of LHASIoV related to utilizing the benefit of attestation messages and minimizing their numbers. All in all, the results of our experiments confirm the scalability and computational practicality of LHASIoV from many points of view.

VII. CONCLUSION AND FUTURE WORK

In this work, we presented a novel Location-aware and Healing attestation scheme for Air-supported Internet of Vehicles (LHASIoV) to secure the IoV network from various internal and external security threats. LHASIoV works in two phases: (i) registration and key generation phase, performed in a secured environment by a Trusted Authority (TA) to initially deploy the network devices, and (ii) attestation, done periodically by the TA to check (and heal) software integrity in different network devices (i.e., roadside units, flying objects, and vehicles). We provide a proof-of-concept implementation for LHASIoV which is based on the TrustLite security architecture, and we also present the comparisons of LHASIoV against related state-of-the-art schemes

to show its effectiveness. Finally, we proved and assessed the practical feasibility of LHASIoV by implementing it in Omnetpp simulation tool. All the performance evaluations done in terms of various metrics such as resilience to different security attacks, attestation overhead, communication/storage/computational costs, attestation delay, mobility, and scalability, show the correctness and effectiveness of our proposed solution and its superiority over the existing approaches. Our work is an adequate starting point for researchers to provide security in heterogeneous IoV scenarios.

In the future, we plan to extend our proposed protocol to have several layers of functionalities that allow adding new types of components to the network system. This would be in line with the rapid growth of technology in IoV industry. We also plan on evaluating our proposed protocol on the physical testbed. It will help to get us new insights such as how much it modifies the hardware in current IoV scenarios and also how costly these changes are. This is required information for vehicle manufacturers as they will prefer low-cost solutions.

REFERENCES

[1] V. Hassija, V. Chamola, G. Han, J. J. P. C. Rodrigues, and M. Guizani, "DAGIoV: A framework for vehicle to vehicle communication using directed acyclic graph and game theory," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4182–4191, Apr. 2020.

[2] T. Alladi, V. Chamola, B. Sikdar, and K. R. Choo, "Consumer IoT: Security vulnerability case studies and solutions," *IEEE Consum. Electron. Mag.*, vol. 9, no. 2, pp. 17–25, Mar. 2020.

[3] T. Alladi, S. Chakravarty, V. Chamola, and M. Guizani, "A lightweight authentication and attestation scheme for in-transit vehicles in IoV scenario," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14188–14197, Dec. 2020.

[4] M. Chen, Y. Tian, G. Fortino, J. Zhang, and I. Humar, "Cognitive Internet of Vehicles," *Comput. Commun.*, vol. 120, pp. 58–70, May 2018.

[5] X. Li et al., "Cognitive AmBC-NOMA IoV-MTS networks with IQI: Reliability and security analysis," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 2596–2607, Feb. 2023.

[6] Q. Wang, X. Chen, X. Jin, X. Li, D. Chen, and X. Qin, "Enhancing trustworthiness of Internet of Vehicles in space–air–ground-integrated networks: Attestation approach," *IEEE Internet Things J.*, vol. 9, no. 8, pp. 5992–6002, Apr. 2022.

[7] J. B. Kenney, "Dedicated short-range communications (DSRC) standards in the United States," *Proc. IEEE*, vol. 99, no. 7, pp. 1162–1182, Jul. 2011.

[8] J. Lee, G. Kim, A. K. Das, and Y. Park, "Secure and efficient honey list-based authentication protocol for vehicular ad hoc networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 3, pp. 2412–2425, Jul. 2021.

[9] C. Zhou et al., "Deep reinforcement learning for delay-oriented IoT task scheduling in space-air-ground integrated network," 2020, *arXiv:2010.01471*.

[10] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, "Connected vehicles: Solutions and challenges," *IEEE Internet Things J.*, vol. 1, no. 4, pp. 289–299, Aug. 2014.

[11] H. Hu, R. Lu, Z. Zhang, and J. Shao, "REPLACE: A reliable trust-based platoon service recommendation scheme in VANET," *IEEE Trans. Veh. Technol.*, vol. 66, no. 2, pp. 1786–1797, Feb. 2017.

[12] L. Vishwakarma, A. Nahar, and D. Das, "LBSV: Lightweight blockchain security protocol for secure storage and communication in SDN-enabled IoV," *IEEE Trans. Veh. Technol.*, vol. 71, no. 6, pp. 5983–5994, Jun. 2022.

[13] M. Rubenstein, A. Cornejo, and R. Nagpal, "Programmable self-assembly in a thousand-robot swarm," *Science*, vol. 345, no. 6198, pp. 795–799, Aug. 2014.

[14] F. Kohnhäuser, N. Büscher, and S. Katzenbeisser, "SALAD: Secure and lightweight attestation of highly dynamic and disruptive networks," in *Proc. Asia Conf. Comput. Commun. Secur.*, May 2018, pp. 329–342.

[15] J. Whitefield, L. Chen, T. Giannetsos, S. Schneider, and H. Treharne, "Privacy-enhanced capabilities for VANETs using direct anonymous attestation," in *Proc. IEEE Veh. Netw. Conf. (VNC)*, Nov. 2017, pp. 123–130.

[16] M. Mustafa, A. M. Buttar, G. S. Sajja, S. Gour, M. Naved, and P. William, "Multitask learning for security and privacy in IoV (Internet of Vehicles)," in *Autonomous Vehicles Volume 1: Using Machine Intelligence*. Hoboken, NJ, USA: Wiley, 2022, pp. 217–233.

[17] K. Mershad, O. Cheikhrouhou, and L. Ismail, "Proof of accumulated trust: A new consensus protocol for the security of the IoV," *Veh. Commun.*, vol. 32, Dec. 2021, Art. no. 100392.

[18] H. Kim, J. Ben-Othman, K.-I. Hwang, and B. Choi, "Intelligent aerial-ground surveillance and epidemic prevention with discriminative public and private services," *IEEE Netw.*, vol. 36, no. 3, pp. 40–46, May 2022.

[19] S. Lee, S. Lee, and H. Kim, "Differential security barriers for virtual emotion detection in maritime transportation stations with cooperative mobile robots and UAVs," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 2461–2471, Feb. 2023.

[20] H. Tan, W. Zheng, and P. Vijayakumar, "Secure and efficient authenticated key management scheme for UAV-assisted infrastructure-less IoVs," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 6, pp. 6389–6400, Jun. 2023.

[21] F. Kohnhäuser, D. Püllen, and S. Katzenbeisser, "Ensuring the safe and secure operation of electronic control units in road vehicles," in *Proc. IEEE Secur. Privacy Workshops (SPW)*, May 2019, pp. 126–131.

[22] A. L. Karn, S. Sengan, D. A. Pustokhin, and I. V. Pustokhina, "Software-defined network-based dynamic access control mechanism for Internet of Vehicles using AdaBoost," *Multimedia Tools Appl.*, pp. 1–25, Oct. 2022.

[23] D. Dolev and A. C. Yao, "On the security of public key protocols," *IEEE Trans. Inf. Theory*, vol. IT-29, no. 2, pp. 198–208, Mar. 1983.

[24] N. Garg, M. Wazid, A. K. Das, D. P. Singh, J. J. P. C. Rodrigues, and Y. Park, "BAKMP-IoMT: Design of blockchain enabled authenticated key management protocol for Internet of Medical Things deployment," *IEEE Access*, vol. 8, pp. 95956–95977, 2020.

[25] A. K. Das and I. Sengupta, "An effective group-based key establishment scheme for large-scale wireless sensor networks using bivariate polynomials," in *Proc. 3rd Int. Conf. Commun. Syst. Softw. Middleware Workshops (COMSWARE)*, Jan. 2008, pp. 9–16.

[26] C. Blundo, A. D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly-secure key distribution for dynamic conferences," in *Proc. Annu. Int. Cryptol. Conf.* Berlin, Germany: Springer, 1992, pp. 471–486.

[27] M. T. Abbas, A. Muhammad, and W.-C. Song, "SD-IoV: SDN enabled routing for Internet of Vehicles in road-aware approach," *J. Ambient Intell. Hum. Comput.*, vol. 11, no. 3, pp. 1265–1280, Mar. 2020.

[28] F. Kohnhäuser, N. Büscher, S. Gabmeyer, and S. Katzenbeisser, "SCAPI: A scalable attestation protocol to detect software and physical attacks," in *Proc. 10th ACM Conf. Secur. Privacy Wireless Mobile Netw.*, Jul. 2017, pp. 75–86.

[29] N. Asokan et al., "SEDA: Scalable embedded device attestation," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2015, pp. 964–975.

[30] P. Koeberl, S. Schulz, A.-R. Sadeghi, and V. Varadharajan, "TrustLite: A security architecture for tiny embedded devices," in *Proc. 9th Eur. Conf. Comput. Syst.*, Apr. 2014, pp. 1–14.

Mohamed A. El-Zawawy received the B.Sc. degree in computer science and the M.Sc. degree in computational science from Cairo University in 1999 and 2002, respectively, and the Ph.D. degree in computer science from the University of Birmingham in 2007. From 2007 to 2014, he was an Assistant Professor of computer science with the Faculty of Science, Cairo University, where he was an Associate Professor of computer science. In 2009, he was an extra-ordinary Senior Researcher with the Institute of Cybernetics, Tallinn University of Technology, Estonia. He was a Teaching Assistant with Cairo University from 1999 to 2003 and Birmingham University from 2003 to 2007. He has been a Full Professor of computer science with the Faculty of Science, Cairo University, since 2022. His current research interests include the security and privacy of android and the IoT. In this area, he has published many articles in topmost international peer-reviewed journals.

Chhagan Lal received the Ph.D. degree in computer science and engineering from the Malaviya National Institute of Technology, Jaipur, India, in 2015. He was a Post-Doctoral Fellow with the Department of Mathematics, University of Padua, Italy, where he was a part of the SPRITZ Research Group. He was a Post-Doctoral Research Fellow with the Simula Research Laboratory, Norway. He is currently a Senior Researcher with the Department of Intelligent Systems, Cybersecurity Group, TU Delft, The Netherlands. His current research interests include the applications of blockchain technologies and smart contracts, network security, software-defined networking, and the securing Internet of Things networks. He received the Canadian Commonwealth Scholarship under the Canadian Commonwealth Scholarship Program to work at the University of Saskatchewan, Saskatoon, SK, Canada, during the Ph.D. degree.

Mauro Conti (Fellow, IEEE) received the Ph.D. degree from the Sapienza University of Rome, Italy, in 2009. After his Ph.D. degree, he was a Post-Doctoral Researcher with Vrije Universiteit Amsterdam, The Netherlands. In 2011, he joined as an Assistant Professor with the University of Padua, Italy, where he became an Associate Professor in 2015 and a Full Professor in 2018. He has been a Visiting Researcher with GMU, UCLA, UCI, TU Darmstadt, UF, and FIU. He is currently a Full Professor with the University of Padua. He is also affiliated with TU Delft and the University of Washington, Seattle. His research is funded by companies, including Cisco, Intel, and Huawei. His current research interests include security and privacy. In this area, he has published more than 450 papers in topmost international peer-reviewed journals and conferences. He has been awarded with the Marie Curie Fellowship by the European Commission in 2012 and the Fellowship by the German DAAD in 2013. He is a Senior Member of ACM and a fellow of the Young Academy of Europe. He was the Program Chair of TRUST in 2015, ICISS in 2016, WiSec in 2017, ACNS in 2020, and CANS in 2021. He was the General Chair of SecureComm in 2012, SACMAT in 2013, NSS in 2021, and ACNS in 2022. He is the Editor-in-Chief of IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, an Area Editor-in-Chief of IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, and an Associate Editor of several journals, including IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, and IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT.