EASBF: An Efficient Authentication Scheme over Blockchain for Fog Computing-enabled Internet of Vehicles

Merzougui Salah Eddine, Mohamed Amine Ferrag, Othmane Friha, Leandros Maglaras

Abstract—The concept of the Internet of Vehicles (IoV) has attracted the attention of many sectors such as the automotive industry, academia, engineering, etc., offering a wide variety of benefits, such as reduced congestion, safer driving, and less pollution. However, since the deployment of IoT based devices is in an open field, the IoV environments can be a vital issue in the absence of security protections. In this paper, we propose an efficient authentication scheme over blockchain, named EASBF, for secure fog computing-enabled internet of vehicles. Specifically, the proposed EASBF scheme consists of five phases: initialization, registration, mutual authentication, and exchanging keys, consensus, and certificate update. Based on the elliptic curve cryptography, one-way hash function, and blockchain technology, the EASBF scheme can achieve confidentiality, integrity, authenticity, privacy, anonymity, non-repudiation, and perfect forward secrecy. The EASBF scheme is robust against cyber attacks, including, DDoS attack, replay attack, man-in-the-middle attacks, identity theft attack, traffic analysis attack, masquerading attack, and man-in-the-middle attack. Both the AVISPA tool and the random oracle model are used in security analysis. Finally, we demonstrate the effectiveness of the EASBF scheme in terms of two performance metrics, namely, computation overhead, communication overhead, and storage overhead.

Index Terms—Security, Authentication, Internet of vehicles, Fog computing, Blockchain.

I. INTRODUCTION

The development and the emergence of data processing and communication technologies have transformed vehicles into more intelligent units. The integration of detection sensors, communication, and networking capabilities of the vehicles allowed them to interact not only with other vehicles but also with roadside units (RSUs) to share information smartly in real-time. The opportunity to share information between them and other entities has allowed the emergence of vehicular ad hoc networks (VANETs). These urban vehicle networks guarantee a wide variety of services that are generally aimed at the vehicles and their users in particular, such as leisure services, applications for road safety, intelligent traffic control, and others [1]. To make intelligent decisions, the vehicular networks cannot process massive amounts of data since they are not able to process, analyze, and aggregate or evaluate information collected by other vehicles and network infrastructure. This situation has created the emergence of the Internet of Vehicles (IoV), which is one of the revolutionary domains of the Internet of Things (IoT) [2]–[4].

The IoV uses network technologies, including Cloud and Fog Computing, to integrate intelligent units such as humans and devices alongside vehicles and road side units [5]. Moreover, with the rise of intelligent vehicles on the global market, a massive amount of in-vehicle equipment has been provided, which collects, manages, and exchanges massive amounts of data, which has resulted in a very large accumulation of traffic on the network [7]. The concept of Vehicle Fog Calculation (VFC) has been integrated to improve the efficiency of communication and minimize latency when collecting, processing, organizing, and storing real-time traffic data [8], [9]. It improves communication efficiency and minimizes the latency while gathering, processing, arranging, and preserving the traffic data in real-time. Many previous studies have shown that IoV networks still face many difficult threats, particularly security and privacy issues, such as illegal data injection, information leakage, lack of security key update management mechanisms, location data spoofing, data replay, denial of service, data tampering, repudiation, etc [10]–[14]. Security threats in the IoV environment are not just about potential business losses, it could also include human casualties. Besides, IoV involve specific issues and challenges, such as the distribution of keys, ensuring privacy, portability and limited error tolerance [15].

In order to attack both the privacy and safety of drivers and passengers, the attackers in the IoV can lunch many attacks, which can cause security and privacy issues such as data forgery, identity spoofing, and disclosure of sensitive information. For this reason, a great deal of research on security and privacy enhancement methods has been proposed [15]–[17]. In addition to personal security and privacy-preservation, there are other security objectives such as confidentiality of information, mutual authenticity, integrity, anonymity, etc. To provide these requirements, blockchain has emerged as an appropriate technology for decentralized applications and complex vehicle network environments [18]. While relying on the offered security of this emerging technology, data is protected and cannot be altered by attackers through specialized cryptographic features offered such as hash functions [19]–[21].
In this paper, in order to achieve authentication inside fog computing-enabled internet of vehicles, we propose an efficient authentication scheme over blockchain, named EASBF. The main contributions of this work are summarized as follows:

- Firstly, we propose the EASBF scheme that implements leading-edge cryptographic techniques, such as elliptic curve cryptography, one-way hash function, and blockchain technology, for fog-based IoV-targeted applications.
- Secondly, we provide comprehensive security analysis using the AVISPA tool and random oracle model to prove that the EASBF scheme can achieve security properties, namely, confidentiality, integrity, authenticity, privacy, anonymity, non-repudiation, and perfect forward secrecy. In addition, we analyze the robustness against cyber attacks, including, DDoS attack, replay attack, identity theft attack, traffic analysis attack, masquerading attack, and session key disclosure attack.
- Finally, we demonstrate the effectiveness of the EASBF scheme in terms of two performance metrics, namely, computation overhead, communication overhead, and storage overhead. In addition, we analyze and compare the performance results with related authentication schemes.

The remainder of this paper is organized as follows. We review related work in Section II. In Section III, we review the elliptic curve cryptography, one-way hash function, and blockchain technology. Our proposed EASBF scheme is presented in Section IV followed by security analysis and performance evaluation in Sections V and VI, respectively. Finally, we draw our conclusions in Section VII.

II. RELATED WORK

Research on security and privacy issues on the internet of vehicles has grown tremendously recently. Kang et al. [22] proposed the scheme which uses specialized fog units deployed at the extremity of the network near vehicles called fog pseudonyms, which are composed of road infrastructures to manage nodes’ pseudonyms, but situations with scattered vehicles are not considered. Fan et al. [23] proposed a radio frequency identification solution for mutual authentication scheme based on the cloud to provide effective privacy protection in the IoV system. By keeping the vehicle’s identity anonymous the proposed system protects the owner’s privacy, which prevents attackers that are outside the network from tracking maliciously any exchanged data. However, the confidentiality of the location is not considered and the authentication time is limited.

Chen et al. [15] has proposed an authentication protocol for IoV applications as a solution that addresses many security issues, including identity compromise, identity theft and replay attacks. The authors’ security analysis of this protocol proved that it is safe against these types of attacks. However, other attacks such as DDoS, and man-in-the-middle remain unaddressed. Cui et al. [24] proposed an IoV authentication system that preserves vehicle privacy and resists security attacks. The performance analysis provided, in terms of communication and computation overheads, showed that the proposed system is suitable for IoV requirements. Nevertheless, the protocol is not checked against many essential security features such as perfect forward secrecy, traffic analysis, and session key disclosure. Liu et al. [25] presented an anonymous mutual authentication system, which is used to secure communications between vehicles and RSUs in IoV applications. The system is based on a Certificate-Less Short Signature system (CLSS). Although the authors provided the performance analysis of computational overheads, the communication overheads analysis was not provided.

In [26], the authors presented a broadcast authentication protocol that can function successfully in regions without Roadside units (RSUs). The protocol’s objective is to enhance energy efficiency while ensuring network security in WSN/IoV communications. However, the protocol is not immune to DDoS attacks. In [27], the authors developed a batch authentication system that maintains conditional privacy in the IoV domain using the Elliptic Curve Cryptography (ECC) technique, vehicles and RSUs can authenticate neighboring vehicles in a batch. However, the protocol is not verified against different critical security features such as perfect forward secrecy, and DDoS mitigation. In [28], authors presented two layers Fog-based authentication scheme for IoV named FBIA. The FBIA scheme uses elliptic curve cryptography and random forest algorithm from deep learning patterns using two layers. The first layer is the security authentication layer for vehicles from outside the fog, whereas the second layer is the security surveillance layer for the remaining vehicles. Experimental results showed that the proposed FBIA scheme offers better accuracy and adaptability for a mobile network under a high-speed IoV environment.

In [29], the authors proposed a lightweight mutual authentication and key agreement protocol for IoV applications. The protocol relies on cryptographic hash functions and XOR operations, which results in significantly decreased computational and communication overhead. But, the authors did not verify the protocol against different security attacks such as man in the middle and replay attacks. Ma et al. [30] presented an authentication system based on biometrics for remote diagnosis and maintenance of vehicles in IoV. The authors evaluate the performance of the proposed system using the Scyther formal verification tool, which is insufficient to prove all the claims. In [31], the authors introduced a lightweight batch authentication and verification system, called LABVS, for universal IoV. The LABVS system uses bilinear matching with random numbers, a secret key, concatenation, and XOR operations, which can result in efficient computation time and effective protection against a variety of attacks. In [32], the authors have designed a lightweight communication protocol for the different components of the IoV. The protocol lacks various proofs of essential security features such as non-traceability, non-repudiation, DDOS mitigation and traffic analysis. In [33], the authors proposed an authentication protocol in the IoV. The protocol uses Physical Unclonable Functions to provide desirable security features, but masquerading and man in the middle attacks have not been proven.
A researcher named "Satoshi Nakamoto" has proposed and applied the idea of the blockchain in 2008 to implement the Bitcoin crypto-currency [34]. After that, several works emerged the blockchain technology in the field of IoV to ensure decentralization, traceability, durability, reliability and integrity, and other security objectives. Wang et al. [35] proposed a scheme using an IoV architecture based on Blockchain technology. They rely on the Byzantine consensus algorithm to ensure the authentication and the Gossip Protocol to complete the communication, but it’s not feasible for a large scale system. Also, the authors did not provide formal security analysis for the proposed protocol. Yao et al. [36] proposed a lightweight mechanism for anonymous authentication that uses the blockchain technology as an assist named BLA dedicated to the provision of vehicular fog services in a distributed way. These services are provisioned to driving vehicles. The BLA scheme combines both modern cryptographic and blockchain technologies to ensure anonymity, and grants drivers the responsibility to preserve their privacy. Nevertheless, diverse security features were not discussed nor proved, e.g. masquerading resistance, session key disclosure resistance, perfect forward secrecy of the protocol, and others.

Cinque et al. [37] proposed a blockchain-based trust management solution for IoV, which is implemented using Hyperledger Fabric. The system showed good overall performance, but protecting the system from possible false praise attacks remained a concern. Zhang et al. [38] introduced a distributed asymmetric group key agreement protocol, named B-AGKA, which is based on the blockchain technology for IoV applications. The B-AGKA protocol proposed that users authenticate each other before agreeing on a group key to protecting against illegal participation. However, diverse security features were not proved nor discussed, e.g. perfect forward secrecy of the protocol, man in the middle resistance, and others.

III. PRELIMINARIES

In this section, we review the elliptic curve cryptography, one-way hash function, and blockchain technology, which will serve as the basis of the proposed EASBF scheme.

A. Elliptic Curve Cryptography

An elliptical curve formed of a set of points.

A specific mathematical equation is used to produce this curve, which is presented by the following equation:

\[ y^2 = x^3 + Ax + B \]  

where the discriminant of \( x^3 + Ax + B \) is not zero.

\[ \Delta = -(4A^3 + 27B^2) \neq 0 \]

Note that this curve includes the point O at infinity.

**Definition 1** (ECDLP: Elliptic Curve Discrete Logarithm Problem). Against two arbitrary and precise points \( Q, P \), compute a scalar \( x \) such that:

\[ Q = xP \]  

Where the elliptical operation \( xP \) signifies that the point \( P \) on the elliptical curve is added to itself \( x \) times. The advantage of an adversary \( A \) to calculate \( x \) during polynomial time \( t \) is as given:

\[ \text{Adv}^ECDLP_A(t) = Pr_b[A(Q, P) = x] \]

\[ \text{Adv}^ECDLP_A(t) \leq \epsilon, \text{ is the ECDLP assumption concluded,} \]

where \( \epsilon > 0 \) very is small [39].

B. One-way Hash Function

A one-way hash function is simply a function that receives a message of a variable length as its input and delivers a string with a fixed length as its output. The output is called the hash code of the input. This hash is used to check the integrity of the messages exchanged via the network.

**Definition 2** (Collision resistance characteristics). Secure, collision resistant hash function, predefined \( Hf() \). The advantage of an adversary \( A \) to find a couple \( (m1 \neq m2) \) so that \( (Hf(m1) = Hf(m2)) \) is marked as \( \text{Adv}^HAsh_A(t) = Pr_b[(m1,m2)\leftrightarrow A : (m1 \neq m2), (Hf(m1) = Hf(m2))] \), where \( A \) is permitted to choose \( (m1,m2) \) arbitrarily in polynomial time \( t \). The collision resistance deduce that \( \text{Adv}^HAsh_A(t) \leq \epsilon \) [39].

![Fig. 1: Blockchain structure.](image)

C. Blockchain Technology

The infrastructure of blockchain technology is presented as a multi-user system, where users are distributed over a network, and share private information, and exchange results of authentications in other regions. Those users agree on an exchange protocol called consensus algorithm instead of relying on an intermediary among them, which helps them achieve mutual trust and allows them to validate peer-to-peer transactions between them [40]-[42]. To build the blocks of a blockchain-based system, the following three technologies are used, namely, a consensus mechanism, cryptography hash functions, and digital signatures, as shown in Fig. 1.

1) **Consensus Algorithm**: In 1999 Miguel Castro and Barbara Liskov, have proposed a consensus algorithm for byzantine faults called The Practical Byzantine Fault Tolerance algorithm (PBFT) [43]. This algorithm provides properties of trustworthiness and robustness in a synchronized environment. To tolerate the simultaneous byzantine faults it requires...
In our work, we use the PBFT algorithm on the internet of vehicles network [44]. This technology can mitigate the attack of tampering of the database managed by a central authority, and also it is highly suitable for delay-sensitive IoV applications [36]. These characteristics encouraged us to explore the application of blockchain technology to build our authentication scheme.

D. Fog Computing

A key requirement for the IoV is security with strict latency [33]. Legacy data-center authentication models are not suited to the fast-moving vehicles, as these models often either ignore user privacy or disregard the time required for driving the vehicles [36]. Instead of performing the entire processing at the cloud center, Fog computing can supplement some of it at the edge of the network with any device with storage, computing, and network connectivity, to collect data associated with the IoV application.

E. Threat models

We consider that the communication channel in between system components is public and insecure. An adversary \( A \) can listen to, modify, forge, or delete the content of the transmitted messages. Specifically, we consider that the adversary \( A \) can perform the following categories of attacks, including DDoS attacks, replay attacks, man-in-the-middle attacks, identity theft attacks, traffic analysis attacks, and masquerading attacks [45], [46].

- **DDoS attacks**: The objective of denial of service (DoS) attacks is to make the authentication service unavailable, by (1) overwhelming the authentication service with an enormous amount of traffic to make it busy, so that it is unable to provide access to legitimate users, or (2) changing a legitimate user’s authentication information to false data. This type of attack mainly uses the limits of bandwidth and transmission power to bring down the IoV system, since most of the major IoV components are exposed outdoors and have poor protection, so it is easy to interfere with, controlled, or destroyed.

- **Replay attacks**: It consists of intercepting data packets between legitimate parties and relaying them to their destination without modification. Its objectives in IoV infrastructure are to intercept data packets between IoV nodes and then forward them to their final destination without any alterations to trigger malicious actions or cause an unknown system state.

- **Man-in-the-middle (MITM) attacks**: An adversary \( A \) can secretly relay and even alter the communication between two parties by impersonating them, believing that they are communicating directly, but in fact, the entire conversation is under the control of the adversary \( A \). In the IoV system, MITM attacks can cause serious damage not only to properties but also to human lives, as adversary \( A \) can provide false information that can lead to dangerous situations such as theft or even death.

- **Identity theft attacks**: The adversary \( A \) covers up under a false identity by using an alleged authenticity from a legitimately authenticated IoV node, broadcasting false and harmful messages, or carrying out malicious attacks.

- **Traffic analysis attacks**: It is based on the perception of the adversary \( A \) on the network. In this type of attack, the adversary \( A \) does not have to compromise the data. Instead, \( A \) simply eavesdrops on network traffic to find vital information such as the location of key nodes, routing patterns, and application behavior patterns.

- **Masquerading attacks**: The adversary \( A \) can obtain critical information by pretending to be a legitimate node. In IoV, masquerade attacks can bring deadly threats to the network while attackers can conceal their identities with spoofed objects.

- **Session-key compromise**: If the adversary \( A \) can manage to compromise the session key, every piece of encrypted data will be available for him/her. \( A \) will be able to read, write, delete, and forge messages, terminate the session, and many other malicious actions.

IV. PROPOSED EASBF SCHEME

As shown in Fig. 2, the system model of the EASBF scheme consists of seven types of nodes, namely, Road side unit, On-board unit, Trusted Authority, Certification Authority, Blockchain Manager, Authentication Manager, and Fog Area.

- **Road side unit (RSU)**: In IoVs, RSUs are the nodes in charge of carrying out communications. They supply vehicles through different services and relay different traffic information between the other nodes. In the scenario envisaged, the RSUs provide services to legitimate users and help the vehicles to communicate with the IoV infrastructure.

- **On-board unit (OBU)**: OBUs are the units and sensors installed in vehicles to make different interactions. We use three types of interactions, namely, V2V, V2I, and V2R. V2V is interaction with other vehicles. V2I is the interaction with the infrastructure, i.e. RSUs. V2R are the interactions with the environment/road. Therefore, the OBU is supplied by communication, computation, and storing entities. In our scenario, OBUs refers to the vehicles/users who must be registered and authenticated to access the services provided.

- **Trusted Authority (TA)**: The TA is a central secured authority that has the responsibility of registering the OBUs and RSUs nodes. In addition, the TA is responsible for initializing and publishing public parameters for the cryptographic functions used.

- **Certification Authority (CA)**: The CA is also a trusted entity, which is responsible for updating the certifications in each Fog area. It can be a parking space, a special RSU, a service station, or a dedicated drone.

- **Blockchain Manager (BM)**: The BM is essentially responsible for the management of the Blockchain and the authentication of OBUs. The BM is proposed to create a trusted connection with each other vehicles and Fog infrastructure in a particular Fog area. This entity is registered and authorized by the TA.

- **Authentication Manager (AM)**: AMs help to write down the results of the authentication in the large public reg-
### A. Description of EASBF scheme

The EASBF scheme consists of five phases: Initialization, Registration, Mutual authentication, and exchanging keys, Consensus, and Certificate update. Tab. I summarizes some abbreviations and notations used in this work.

1) **Initialization:** During the initialization, the TA establishes the environment for the next phases of the proposed scheme by producing different public parameters. This phase is developed in the following three main steps:

- **Step 1:** The TA generates the elliptical curve $E$ and sets its public parameters $n$ and $P$.
- **Step 2:** The TA calculates its public and secret keys, including, $(SK_{TA}$ and $PK_{TA})$, where a secret key is generated as follows:

$$SK_{TA} \in \mathbb{Z}_p$$

Then it calculates the public key as:

$$PK_{TA} = SK_{TA} \cdot P$$

- **Step 3:** The TA defines a unidirectional function $H_f()$, which is used to check the integrity during phase 3. Then, the parameters $<E, P, n, H_f(), PK_{TA}>$ are published publicly.

2) **Registration:** For the registration phase of three nodes, namely, $TA$, $OBU_i$, and $RSU_k$, the EASBF scheme invokes Algorithms [1][2] and [3]. During the registration, OBUs and RSUs are registered nearby TA. Their respective identities $(ID_{OBU}$ and $ID_{RSU})$ are kept anonymous and never exchanged without getting ciphered. The registration phase runs the following steps:

- **Step 1:** The OBU chooses his identity $ID_{OBU}$ and generates $TW_{OBU}$, which is the time window at the same time. The time window will be used in the validation of the transmitted messages between him and the TA in order to make sure that they are not delayed or transmitted in the future.
- **Step 2:** The OBU creates a temporary token composed of the time window and its identity as follows: $To_0 = \{ID_{OBU} \mid TW_{OBU}\}$. Then, the OBU uses the public key of TA $PK_{TA}$ to encrypt it. The encrypted token $CPK_{TA} (To_0)$ and its hash value $h_0 = H_f(To_0)$ are sent to the TA.
- **Step 3:** When receiving the message $(CPK_{TA} (To_0), h_0, TW_{OBU})$, the TA decrypts it using its secret key $SK_{TA}$, then calculates $h_0' = H_f(To_0)$ to compare it with $h_0$. If they are equal, then it goes to the next steps, otherwise, it requests the OBU to retry sending the message. The TA extracts the values $ID_{OBU}, TW_{OBU}$ and $PK_{OBU}$. Then, the TA validates token by checking its time window. If $TW_{OBU}$ doesn’t exceed the allowed time window, then the TA proceeds to the next steps, otherwise the transmission is interrupted and interrupt the connections. Next, the TA checks in its Blockchain network whether if $ID_{OBU}$ is available or not. If it is available, the TA asks the $i^{th}$ OBU to choose a new identity and re-do the steps above.
- **Step 4:** The TA produces a secret key and calculates its public key $To_1 = PK_{OBU}, SK_{OBU} >$ for the OBU and a signature $H_f(SK_{TA} \mid ID_{OBU})$ and a time window $TW_{TA}$, encrypting them using the initial public key of OBU $PK_{OBU}$ then calculates its hash $h_1 = H_f(To_1, H_f(SK_{TA} \mid ID_{OBU}), TW_{TA})$ and transmits $CPK_{OBU} (To_1, H_f(SK_{TA} \mid ID_{OBU}), TW_{TA})$ and $h_1$ to $OBU_i$. In addition, the TA also calculates and stores the value of $Cert = H_f(SK_{OBU} \mid ID_{OBU}) \bigoplus H_f(SK_{TA} \mid ID_{OBU})$ in its distributed register and transmits also $S = H_f(SK_{TA} \mid ID_{OBU})$ to the $OBU_i$.
- **Step 5:** The $i^{th}$ OBU decrypts the message using its initial secret key $SK_{OBU}^i$, validate the time window $TW_{TA}$, then calculates $h_1' = H_f(To_1, H_f(SK_{TA} \mid ID_{OBU}), TW_{TA})$ and compare it with $h_1$ if they are equal, its integrity is valid and the OBU stores the values $PK_{OBU}, SK_{OBU}$ and $S = H_f(SK_{TA} \mid ID_{OBU})$ in its repository otherwise he asks the TA to retry sending the last transmitted message.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E$</td>
<td>Elliptic curve</td>
</tr>
<tr>
<td>$P$</td>
<td>Base point of curve $E$</td>
</tr>
<tr>
<td>$n$</td>
<td>A prime number</td>
</tr>
<tr>
<td>$PK_{X}$</td>
<td>Public key of $X$</td>
</tr>
<tr>
<td>$SK_{X}$</td>
<td>Secret key of $X$</td>
</tr>
<tr>
<td>$H_f()$</td>
<td>One way hash function</td>
</tr>
<tr>
<td>$ID_{X}$</td>
<td>Identifier of $X$</td>
</tr>
<tr>
<td>$TW_{X}$</td>
<td>Time window generated by $X$</td>
</tr>
<tr>
<td>$T_{oi}$</td>
<td>Token $i$</td>
</tr>
<tr>
<td>$h$</td>
<td>Hash value</td>
</tr>
<tr>
<td>$CPK_{X}(M)$</td>
<td>Encrypts message $M$ with the public key of $X$</td>
</tr>
<tr>
<td>$DSK_{X}(M)$</td>
<td>Decrypts message $M$ with the secret key of $X$</td>
</tr>
<tr>
<td>$Cert$</td>
<td>A certificate</td>
</tr>
<tr>
<td>$\oplus$</td>
<td>XOR operation</td>
</tr>
<tr>
<td>$S$</td>
<td>A signature</td>
</tr>
<tr>
<td>$rn_i$</td>
<td>A random number $\in \mathbb{Z}_P$</td>
</tr>
<tr>
<td>$N_i$</td>
<td>A constructed number of $rn_i$</td>
</tr>
<tr>
<td>$AuthTok_{X}$</td>
<td>An authentication token generated by $X$</td>
</tr>
<tr>
<td>$SK_{oi}$</td>
<td>A session key</td>
</tr>
<tr>
<td>$CDK(f)$</td>
<td>Cryptographic key derivation function</td>
</tr>
<tr>
<td>$V_{req}$</td>
<td>A vote request token</td>
</tr>
<tr>
<td>$B_i$</td>
<td>Index of block $i$</td>
</tr>
<tr>
<td>$Block$</td>
<td>Block content</td>
</tr>
<tr>
<td>$AM$</td>
<td>Identifier of the AM marked speaker</td>
</tr>
<tr>
<td>$SignK(B)$</td>
<td>Sign $B$ with key $K$</td>
</tr>
<tr>
<td>$V_{req}$</td>
<td>A vote response token</td>
</tr>
<tr>
<td>$Acc_{req}$</td>
<td>A Fog service access request token</td>
</tr>
<tr>
<td>$Maj_{req}$</td>
<td>A certificate update request token</td>
</tr>
<tr>
<td>$A$</td>
<td>Attacker</td>
</tr>
</tbody>
</table>
Algorithm 1 Registration phase: TA
1: Receives from OBU/RSU:
2: CPKTA(T0), h0, TW0
3: h0’ ← Hf(T0)
4: If (h0’ = h0)
   - Extracts ID0 and TW0
   - Checks the availability and uniqueness of ID0 in its Blockchain
   - Secret key generation: SK0 ∈ Z∗
   - Public key generation: PK0 ← SK0 • P
   - Chooses T0
   - Time window generation: TW0
   - Public key generation: PK0
   - Encryption:
   - Sends to OBU/RSU:
   - Else the connection is interrupted

Algorithm 2 Registration phase: OBU/RSU
1: Chooses ID0
2: Time window generation: TW0
3: Generation of a random secret key: SK0 ∈ Z∗
4: Generation of its public key: PK0 ← SK0 • P
5: T0 ← ID0 || TW0 || PK0
6: h0 ← Hf(T0)
7: Encryption: CPKTA(T0)
8: Sends: CPKTA(T0), h0, TW0 to TA

Algorithm 3 Registration phase: OBU / RSU
1: Receives from TA:
2: CPKTA(T0, Hf(SK0 || ID0), TW0, h0, TW0)
3: Checks the validity of TW0
4: h0’ ← Hf(T0, Hf(SK0 || ID0), TW0)
5: If (h0’ = h0)
   - S ← Hf(SK0 || ID0)
   - Saves: PK0, SK0 and S
   - Sends to OBU / RSU:
6: Else asks the TA to retry sending the last transmitted message.

3) Mutual authentication and exchanging keys: In this phase, OBUs exchange keys and mutually authenticate with BMs and in the end, they share a session key for a subsequent connection. The mutual authentication and exchanging keys phase invokes Algorithms 4, 5 and 6, by running the following steps:

Algorithm 4 Authentication phase: OBU
1: Generates a random number r1 ∈ Zp
2: Time window generation: TW0
3: N1 ← r1 • P • SK0
4: T0 = Hf(SK0 || ID0 || TW0 || T0)
5: AuthTok = Hf(N1 || ID0 || TW0 || T0)
6: CPKBM(AuthTok, r1, ID0, TW0)
7: Sends to BM:
   - Step 1: Initially the ith OBU generates a random number r1 and calculates N1 by performing multiplicative operations ECC (r1 • P • SK0). Then, it generates a time window TW0.
- Step 2: The OBU uses the hash function $H_f()$, concatenation and XOR operations to calculate the value of the token $To_{OBU} = H_f(SK_{OBU} \parallel ID_{OBU}) \oplus S$; as shown in algorithm 4. Then builds an authentication token as follows:

$$AuthTok_{OBU} = H_f(N_1 \parallel ID_{OBU} \parallel TW_{OBU} \parallel To_{OBU})$$

Then encrypt $AuthTok_{OBU}$, $rn_1$, $ID_{OBU}$, $TW_{OBU}$ using the public key $PK_{BM_j}$ of $BM_j$ as follows: $CPK_{BM_j} (AuthTok_{OBU}, rn_1, ID_{OBU}, TW_{OBU})$.

Finally, the values $CPK_{BM_j} (AuthTok_{OBU}, rn_1, ID_{OBU}, TW_{OBU})$, $TW_{OBU}$ > are transmitted to the $j^{th}$ BM for later analysis.

**Algorithm 5 Authentication phase: $BM_j$**

1. Receives from $OBU_i$:
   $CPK_{BM_j}(AuthTok_{OBU}, rn_1, ID_{OBU}, TW_{OBU}, TW_{OBU})$

2. Decryption: $CPK_{BM_j} (AuthTok_{OBU}, rn_1, ID_{OBU}, TW_{OBU})$

3. Checks the validity of: $TW_{OBU}$

4. Search the blockchain for the value $Cert$ corresponding to the value of $ID_{OBU}$

5. $AuthTok^*_{OBU} = H_f(rn_1 \bullet PK_{BM_j} \parallel ID_{OBU} \parallel TW_{OBU} \parallel Cert)$

6. If ($AuthTok_{OBU} = AuthTok^*_{OBU}$)
   - $OBU_i$ is marked authentic
   - Generation of a random number $rn_2 \in Z_p$
   - $N_2 \leftarrow rn_2 \ast P \ast SK_{BM_j}$
   - Time window generation: $TW_{OBU}$
   - $AuthTok_{BM_j} = H_f(N_2 \parallel TW_{BM_j} \parallel Cert)$
   - $SK_i = CDKf(ID_{OBU} \parallel Cert \parallel TW_{OBU} \parallel TW_{BM_j})$
   - $CPK_{BM_j} (AuthTok_{BM_j}, rn_2, ID_{OBU}, TW_{OBU})$
   - $CPK_{OBU} (AuthTok_{BM_j}, rn_2, TW_{BM_j})$
   - Sends to $OBU_i$:
   $CPK_{OBU} (AuthTok_{BM_j}, rn_2, TW_{BM_j})$, $TW_{BM_j}$

7. Else The connection is interrupted

- Step 3: When the $BM_j$ receives these values, it firstly decrypts $CPK_{BM_j} (AuthTok_{OBU}, rn_1, ID_{OBU}, TW_{OBU})$ using its private key $SK_{BM_j}$, then it validates the $TW_{OBU}$ time window. And if the validation is successful, the BM will therefore search for the $Cert$ value corresponding to the value of $ID_{OBU}$ in the OBU list recorded on the Blockchain. Using these values, BM calculates an authentication token to validate the veracity of $AuthTok_{OBU}$ as follows:

$$AuthTok^*_{OBU} = H_f(rn_1 \bullet PK_{OBU} \parallel ID_{OBU} \parallel TW_{OBU} \parallel Cert)$$

If the values of $AuthTok_{OBU}$ and $AuthTok^*_{OBU}$ correspond, then the BM validates the $j^{th}$ OBU and marks it as authentic, otherwise it interrupts the connection.

- Step 4: In this step, $BM_j$ chooses his random number $rn_2$ to calculate $N_2 = rn_2 \ast P \ast SK_{BM_j}$, then generates his time window $TW_{BM_j}$, and finally he calculates his $AuthTok_{BM_j}$ as follows:

$$AuthTok_{BM_j} = H_f(N_2 \parallel TW_{BM_j} \parallel Cert)$$

Then encrypt $AuthTok_{BM_j}$, $rn_2$, $TW_{BM_j}$ using the public key $PK_{OBU}$ of $OBU_i$ as follows: $CPK_{OBU} (AuthTok_{BM_j}, rn_2, TW_{BM_j})$. Also, it calculates a symmetric key, using the session-key derivation function CryptDeriveKey (CDKf), to ensure the security without re-authenticating for the upcoming communication sessions between these two nodes:

$$SK_{ij} = CDKf(ID_{OBU} \parallel Cert \parallel TW_{OBU} \parallel TW_{BM_j})$$

Finally, it relays the values $CPK_{OBU} (AuthTok_{BM_j}, rn_2, TW_{BM_j})$, $TW_{BM_j}$ to the $i^{th}$ OBU.

**Algorithm 6 Authentication phase: $OBU_i$**

1. Receives from $BM_j$:
   $CPK_{OBU} (AuthTok_{BM_j}, rn_2, TW_{BM_j})$, $TW_{BM_j}$

2. Decryption: $CPK_{OBU} (AuthTok_{BM_j}, rn_2, TW_{BM_j})$

3. Checks the validity of $TW_{BM_j}$

4. $AuthTok^*_{BM_j} = H_f((rn_2 \ast PK_{BM_j}) \parallel TW_{BM_j} \parallel To_{OBU})$

5. If ($AuthTok_{BM_j} = AuthTok^*_{BM_j}$)
   - $BM_j$ is marked authentic
   - $SK_{ij} = CDKf (ID_{OBU} \parallel TW_{OBU} \parallel TW_{BM_j} \parallel TW_{BM_j})$

6. Else The connection is interrupted.

- Step 5: Upon receiving the values $CPK_{OBU} (AuthTok_{BM_j}, rn_2, TW_{BM_j})$, $TW_{BM_j}$, the OBU decrypts $CPK_{OBU} (AuthTok_{BM_j}, rn_2, TW_{BM_j})$ using its private $SK_{OBU}$ key, and checks if the time window is valid then proceeds.

- Step 6: The OBU collects data related to the $BM_j$ and computes $AuthTok_{BM_j}$ to validate the authenticity of the BM as follows:

$$AuthTok^*_{BM_j} = H_f(rn_2 \ast PK_{BM_j} \parallel TW_{BM_j} \parallel To_{OBU})$$

If $AuthTok^*_{BM_j}$ and $AuthTok_{BM_j}$ are equal this signifies the authenticity of the $j^{th}$ BM; this implies that the two parties have performed a mutual authentication and are ready to transmit different data between them.

- Step 7: In the end, the $OBU_i$ is able to calculate the symmetric key and stores it for further communications:

$$SK_{ij} = CDKf (ID_{OBU} \parallel TW_{OBU} \parallel TW_{BM_j} \parallel TW_{BM_j})$$

4) Consensus: In the proposed scheme, the results of the authentication are transferred to the Blockchain. We consider a PBFT consensus algorithm to form the large public register. We assume there are $k$ number of AM with the possibility of writing a block in the large register. As shown in Fig. 3 one of the AMs plays the role of a “Speaker” which is responsible for starting the process of the consensus; while the others act as members of the congress who participate in the voting mechanism initiated by the Speaker. To save time and avoid selecting many speakers, we give a selected Speaker the ability to lead several cycles of the consensus but we deny
it from participating the vote. The consensus phase runs the following steps:

- **Step 1:** Select speaker $S$ using the following evaluation:
  
  \[ S = (B_i \mod k) + 1 \]

  Where $B_i$ refers to the current block index.

- **Step 2:** After the successful authentication and the exchange of keys between the $OBU_i$ and $BM_j$, the $j^{th}$ BM shares the authentication results with all the AMs. Next, the AMs store the results after receiving them; and then transfer them to the large public register.

- **Step 3:** The voting process starts right after the creation of the block that contains the results of the authentication after $t$ time needed to generate a block. In round one, the Speaker sends a request $<Vote_{req}, B_i, AM_S, Block, \text{SignAM}_S(\text{Block})>$ via a Fog node (RSU) to the members of the congress asking them to start voting. Here, the variable $Vote_{req}$ denotes the request of the speaker to the other members of the congress to start the vote, the variable $B_i$ denotes the index of the block created, the variable $AM_S$ denotes the identifier of the AM marked speaker, the variable $Block$ denotes the block created itself and finally the variable $\text{SignAM}_S(\text{block})$ denotes the signature of the block created with the key $AM_S$.

- **Step 4:** The request is transmitted as a Pre-Prepare message to the members of the congress via the Fog node. A response is sent from the correct replicas as a Prepare message to all other members.

- **Step 5:** The members of the congress agreed on the request of the speaker once they have received $2f$ Prepare messages from other members and the associated Pre-Prepare. Therefore they send to all other members a Validation message.

- **Step 6:** After receiving $2f + 1$ associated Validation, the $k^{th}$ AM shares its vote using $<Vote_{res}, B_i, AM_k, \text{Block}, \text{SignAM}_k(\text{Block})>$ where the variable $Vote_{res}$ denotes the response of the $k^{th}$ AM, the variable $AM_k$ denotes the identifier of the $k^{th}$ AM, the variable $\text{SignAM}_k(\text{Block})$ denotes the signature of the block created with the key $AM_k$.

- **Step 7:** The block containing the authentication results is added to the large register right away after the speaker receives the response from the congress members.

5) **Certificate update:** This particular phase offers vehicles two scenarios, the first is the possibility of moving from one Fog area to another in a transparent manner without having to re-authenticate, and the second is to ask the CA to update its certificate. In the first scenario, the $i^{th}$ OBU sends an $Acc_{req}$ access request consisting of a time window $TW_{OBU_i}$, an encrypted token $CTok = CPK_{BM_i}(ID_{OBU_i}, TW_{OBU_i})$ and $h = Hf(ID_{OBU_i}, TW_{OBU_i})$ to the new $BM_i^*$ via the new $RSU_i^*$. Once the new $BM_i^*$ receives the access request $Acc_{req} = <CTok, TW_{OBU_i}, h^\star>$ and extract the values $CTok, TW_{OBU_i}$ and $h$, it decrypts the token $CTok$ using its secret key $SK_{BM_i}$ to deduce $ID_{OBU_i}$ and $TW_{OBU_i}$, it validates $TW_{OBU_i}$ the time window. Checks the integrity of the message by calculating $h^\star' = Hf(ID_{OBU_i}, TW_{OBU_i})$ and compare it to $h$, if they are both valid, it searches its local database for $Cert$ corresponding to $ID_{OBU_i}$. If there is not a match, it accesses the public register and checks it there. If it finds a match this indicates that the authentication of this vehicle has been already performed in another fog area. After this the $BM_i^*$ accesses the revocation list and checks whether this vehicle’s $Cert$ has been revoked or not, if not, it acknowledge the validity of the $i^{th}$ OBU and $BM_i^*$ directly informs the $RSU_i^*$ to answer request of service sent by $OBU_i$ without the need of a re-authentication. Otherwise, it informs the $RSU_i^*$ to refuse to provide services to this vehicle.

If a unit reports an illegal vehicle during any phase of the process, the TA will retrieve the identity of this vehicle from the large public register and broadcast to all BMS that this vehicle’s public key has become illegal and writes down the $Cert$ corresponding to this vehicle’s identity in the revocation list.

For the second scenario we do the same steps as the first scenario, when the $i^{th}$ OBU enters the CA area it sends $Maj_{req}$ a request to update its certificate to the $CA_j$. This request is formed of a time window $TW_{OBU_i}$, an encrypted token $CTok = CPK_{CA_j}(ID_{OBU_i}, TW_{OBU_i})$ and a hash $h = Hf(ID_{OBU_i}, TW_{OBU_i})$. Once $CA_j$ receive the request $Maj_{req} = <CTok, TW_{OBU_i}, h^\star>$ and extracts the values $CTok, TW_{OBU_i}$ and $h$, it decrypts the token $CTok$ using its secret key $SK_{CA_j}$ to deduce $ID_{OBU_i}$ and $TW_{OBU_i}$, it validates the time window $TW_{OBU_i}$, then checks the integrity of the message by calculating $h^\star' = Hf(ID_{OBU_i}, TW_{OBU_i})$ and compares it to $h$, once they are both valid, it looks for the value $Cert$ corresponding to $ID_{OBU_i}$ in the large public register. If it finds a match, this indicates that this vehicle is authentic, and therefore proceeds to update the certificate. The CA then generates a new pairs of secret-public keys $To^* = PK_{OBU_i}^*$,
SK_{OBU_i} > \text{for the OBU}_i \text{and a time window } TW_{CA_i} \text{ encrypt them using the old public key of the } i^{th} \text{ OBU } PK_{OBU_i} \text{ then calculates the hash } h_1 = H_f(TO^*, TW_{CA_i}). \text{ In addition, the CA calculates and updates the value of Cert corresponding to } I_{DBU_i} \text{ with the new value } \text{Cert} = H_f(SK_{OBU_i} \parallel I_{DBU_i}) \oplus H_f(SK_{CA_i} \parallel I_{DBU_i}) \text{ in the public register and transmits CPK_{OBU} (TO^*, TW_{CA_i}), TW_{CA_i}, h_1 \text{ and } S = H_f(SK_{CA_i} \parallel I_{DBU_i}) \text{ to OBU_i and finally it generates a new session key } SK_{ij}. \text{ The } i^{th} \text{ OBU therefore decrypts the message using its old secret key } SK_{OBU_i}, \text{ validates the time window } TW_{CA_i}, \text{ then calculates } h_1^* = H_f(TO^*, TW_{CA_i}) \text{ and compares it to } h_1 \text{ if they are equal, its integrity is valid and the OBU stores the new values } PK_{OBU_i}, SK_{OBU_i} \text{ and } S = H_f(SK_{CA_i} \parallel I_{DBU_i}) \text{ in its repository and generates its new session key } SK_{ij}.

V. SECURITY ANALYSIS

In this section, we analyze the security properties of our proposed EASBF. Specifically, we provide a formal security verification using the random oracle model as well as the AVISPA tool. Then, we evaluate the security features, including confidentiality, integrity, authenticity, privacy, anonymity, traceability, non-repudiation, and non-interactivity, and perfect forward secrecy. Also, we discuss the resistance against potential attacks, such as DDoS attacks, replay attacks, man-in-the-middle attacks, identity theft attacks, traffic analysis attacks, and masquerading attacks.

A. Formal security verification using Random Oracle Model (ROM)

The following theorems are used to formally prove the security of our scheme against an adversary \( \mathcal{A} \) \cite{39, 47}. Specifically, we define the following oracles:

- **Reveal**: This oracle will unconditionally return the input \( m \) from the associated hash value \( h = H_f(m) \).
- **Extract**: This oracle will returns the scalar \( x \) out of a specified point \( Q = xP \) and \( P \).

**Theorem 1**: Based on the assumption that the secure one-way hash function \( H_f() \) acts like a random oracle, and the hardness hypothesis of the ECDLP, the proposed authentication scheme is secure against an adversary \( \mathcal{A} \) for resolution of: the identity of OBU\( _i \) (I_{DBU_i}), the secret key of TA (SK_{TA}), and the private key of OBU\( _i \) (SK_{OBU_i}).

**Proof** An adversary \( \mathcal{A} \) is given the ability to derive OBU\( _i \)'s I_{DBU_i}, SK_{OBU_i}, and TA's secret key SK_{TA}. To do this, \( \mathcal{A} \) use the Reveal and Extract oracles in the algorithmic experiment EXP_{EASBF}^{ECDLP,HASH}(t) against the proposed authentication scheme EASBF. The probability of success of EXP_{EASBF}^{ECDLP,HASH}(t) is expressed as

\[
Suc_{\mathcal{A},EASBF} = |Prb[EXP_{EASBF}^{ECDLP,HASH}(t) = 1] - 1|.
\]

The advantage of \( \mathcal{A} \) is defined as

\[
Adv_{\mathcal{A},EASBF} = max_{A}(Suc_{\mathcal{A},EASBF} - |Prb[EXP_{EASBF}^{ECDLP,HASH}(t) = 1] - 1|).
\]

\( \mathcal{A} \) is capable of computing I_{DBU_i}, SK_{TA}, and SK_{OBU_i}, if \( \mathcal{A} \) is able to (1) invert the secure one-way hash function \( H_f() \) (2) break the ECDLP. However, according to Definitions 1 and 2, it is computationally infeasible to do so. As a result, \( Adv_{\mathcal{A},EASBF} \leq \epsilon \). Therefore, the proposed authentication scheme is secure against an adversary \( \mathcal{A} \) to derive the identity of OBU\( _i \) (I_{DBU_i}), the private key of OBU\( _i \) (SK_{OBU_i}), and the secret key of TA (SK_{TA}).

**Algorithm 7** EXP_{EASBF}^{ECDLP,HASH}(t)

1: Retrieve TA's public available parameters generated during the initialisation phase: \( \{ E, P, n, H_f(), PK_{TA} \} \), where \( PK_{TA} = SK_{TA} \parallel P \).
2: Call Extract oracle on \( PK_{TA} \) to get \( SK_{TA}' \Leftarrow \text{Extract}(PK_{TA}) \).
3: Eavesdrop the registration messages from OBU\( _i / RSU_k \) to TA: \( \{ CPK_{OBU_i}(TO_i, H_f(SK_{TA} \parallel I_{DBU_i}), TW_{TA_i}), h_1, TW_{TA_i} \} \), where \( TO_i = < PK_{OBU_i}, SK_{OBU_i} > \), and \( h_1 = H_f(TO_i, H_f(SK_{TA} \parallel I_{DBU_i}), TW_{TA_i}) \).
4: Call Reveal oracle on \( h_0 \) and get \( TO_i' \Leftarrow \{ I_{DBU_i} \parallel TW_{DBU_i} \parallel PK_{OBU_i}' \} \). \( \Leftarrow \text{Reveal}(h_0) \).
5: Compute \( TO_i'' = DSK_{TA}'(CPK_{OBU}(TO_i')) \).
6: Compute \( h_0' = H_f(TO_i'') \).
7: If \( (h_0' = h_0) \) Then
   - Accept \( SK_{TA}' \) as the secret key of TA, and I_{DBU_i}' as the identity of OBU\( _i \).
   - Eavesdrop the registration messages from TA to OBU\( _j / RSU_k \): \( \{ CPK_{OBU_j}(TO_j, H_f(SK_{TA}' \parallel I_{DBU_j}), TW_{TA_j}), h_2, TW_{TA_j} \} \), where \( TO_j = < PK_{OBU_j}, SK_{OBU_j} > \), and \( h_2 = H_f(TO_j, H_f(SK_{TA}' \parallel I_{DBU_j}), TW_{TA_j}) \).
   - Call Reveal oracle on \( h_j \) and get \( TO_j' \Leftarrow \{ I_{DBU_j} \parallel TW_{DBU_j} \parallel PK_{OBU_j}' \} \). \( \Leftarrow \text{Reveal}(h_j) \).
   - Call Reveal oracle on \( H_f(SK_{TA}' \parallel I_{DBU_j}), TW_{TA_j} \).
   - Compute \( h_j' = H_f(TO_j', H_f(SK_{TA}' \parallel I_{DBU_j}, TW_{TA_j}) \).
     - If \( (h_j' = h_2) \) then
       - Accept \( TO_j'' = < PK_{OBU_j}', SK_{OBU_j}' \) as the public and the secret keys for OBU\( _j \).
     - else
       - return Fail
     - end If
     - else
       - return Fail
   - end If

**Theorem 2**: Based on the assumption that the secure one-way hash function \( H_f() \) acts like a random oracle, and the hardness hypothesis of the ECDLP, the proposed authentication scheme is invincible against an adversary \( \mathcal{A} \) to compute the secret key of BM\( _j \) (SK_{BM_j}), the authentication token of OBU\( _j \) (AuthTok_{OBU_j}), the authentication token of BM\( _j \) (AuthTok_{BM_j}), and the session key (SK_{ij}).

**Proof** We consider an adversary \( \mathcal{A} \) who has ability to derive BM\( _j \)'s SK_{BM_j} and AuthTok_{BM_j}, OBU\( _j \)’s AuthTok_{OBU_j}, and the session key SK_{ij}. For this, \( \mathcal{A} \) use the Reveal and Extract oracles in the algorithmic experiment EXP_{EASBF}^{ECDLP,HASH}(t) against EASBF, which is provided in Algorithm 8. The probability of success is expressed as

\[
Suc_{\mathcal{A},EASBF} = |Prb[EXP_{EASBF}^{ECDLP,HASH}(t) = 1] - 1|.
\]

The advantage
of $A$ is expressed as $Adv_{ECDLP,\text{HASH}}^{EASBF}(t, q_{rc}, q_{ex}) = \max_{A}(\text{Succ}_{ECDLP,\text{HASH}}^{EASBF})$, where $A$ is capable to make the maximum number of $\text{Reveal}(q_{rc})$ and $\text{Extract}(q_{ex})$ queries. Based on $\text{EXPR}_2^{ECDLP,\text{HASH}}(t)$, $A$ is capable of retrieving $SK_{BM_j}, \text{AuthTok}_{OBU_i}, \text{AuthTok}_{BM_j}, SK_j$. If $A$ is able to (1) invert the secure one-way hash function $H()$ (2) break the ECDLP. Nevertheless, based on Definitions 1 and 2, it is by no means feasible to do so computationally. Consequently $Adv_{ECDLP,\text{HASH}}^{EASBF}(t, q_{rc}, q_{ex}) \leq \epsilon$. Therefore, the proposed authentication scheme is secure against an adversary $A$ to compute the secret key of $BM_j$ ($SK_{BM_j}$), the authentication token of $OBU_i$ ($\text{AuthTok}_{OBU_i}$), the authentication token of $BM_j$ ($\text{AuthTok}_{BM_j}$), and the session key ($SK_j$).

**Algorithm 8** $\text{EXPR}_2^{ECDLP,\text{HASH}}(t)$

1: Eavesdrop the authentication messages from $OBU_i$ to $BM_j$: \{ $CPK_{BM_j}(\text{AuthTok}_{OBU_i}, r_{n_1}, ID_{OBU_i}, \text{TWO}_{OBU_i})$, \text{TWO}_{OBU_i} \}, where $AuthTok_{OBU_i} = H((N_1 \parallel ID_{OBU_i} \parallel \text{TWO}_{OBU_i} \parallel \text{TDO}_{OBU_i}))$, and $\text{TDO}_{OBU_i} = H((SK_{OBU_i} \parallel ID_{OBU_i} \parallel \text{Cert}))$
2: Call $\text{Extract}$ oracle on $PK_{BM_j}$ to get $SK'_{BM_j}$ ← $\text{Extract}(PK_{BM_j})$
3: Compute $DSK'_{BM_j}(CPK_{BM_j}(\text{AuthTok}_{OBU_i}, r_{n_1}, ID_{OBU_i}, \text{TWO}_{OBU_i}))$ to get $\text{AuthTok}'_{BM_j}, r_{n_1}', ID'_{OBU_i}, \text{TWO}'_{OBU_i}$
4: Call $\text{Reveal}$ oracle on $\text{AuthTok}'_{OBU_i}$ and get ($N_1' \parallel ID'_{OBU_i} \parallel \text{TWO}'_{OBU_i} \parallel \text{TDO}'_{OBU_i})$ ← $\text{Reveal}(\text{AuthTok}'_{OBU_i})$
5: Eavesdrop the authentication messages from $BM_j$ to $OBU_i$: \{ $CPK_{OBU_i}(\text{AuthTok}_{BM_j}, r_{n_2}, \text{TWO}_{BM_j})$, \text{TWO}_{BM_j} \}, where $AuthTok_{BM_j} = H((N_2 \parallel \text{TWO}_{BM_j} \parallel \text{Cert}))$
6: Call $\text{Extract}$ oracle on $PK_{OBU_i}$ to get $SK''_{OBU_i}$ ← $\text{Extract}(PK_{OBU_i})$
7: Compute $DSK''_{OBU_i}(CPK_{OBU_i}(\text{AuthTok}_{BM_j}, r_{n_2}, \text{TWO}_{BM_j}))$ to get ($N_2' \parallel \text{TWO}'_{BM_j} \parallel \text{Cert}')$ ← $\text{Reveal}(\text{AuthTok}'_{BM_j})$
8: Call $\text{Reveal}$ oracle on $\text{AuthTok}'_{BM_j}$ and get ($N_2' \parallel \text{TWO}'_{BM_j} \parallel \text{Cert}')$ ← $\text{Reveal}(\text{AuthTok}'_{BM_j})$
9: Compute $\text{AuthTok}''_{OBU_i} = H((r_{n_1}' \parallel PK_{OBU_i}) \parallel ID'_{OBU_i} \parallel \text{TWO}'_{OBU_i} \parallel \text{Cert'})$
10: Compute $\text{AuthTok}''_{BM_j} = H((r_{n_2}' \parallel PK_{BM_j}) \parallel \text{TWO}'_{BM_j} \parallel \text{TDO}'_{OBU_i} \parallel \text{TWO}'_{OBU_i} \parallel \text{TWO}'_{BM_j})$
11: if ((\text{AuthTok}''_{OBU_i} = \text{AuthTok}'_{OBU_i}) AND (\text{AuthTok}''_{BM_j} = \text{AuthTok}'_{BM_j})) then
- Compute $SK_j = CDKf(ID''_{OBU_i} \parallel \text{TDO}'_{OBU_i} \parallel \text{TWO}'_{OBU_i} \parallel \text{TWO}'_{BM_j})$
- Accept $SK'_{BM_j}$ as the secret key of $BM_j$, $SK''_{OBU_i}$ as the secret key of $OBU_i$, $\text{AuthTok}'_{OBU_i}$ as the authentication token of $OBU_i$, $\text{AuthTok}'_{BM_j}$ as the authentication token of $BM_j$, and $SK_j$ as the session key.
else
-return Fail

B. Formal security verification under AVISPA tool

To validate the EASBF proposed scheme, the authentication phase of our system was coded in HLP/S and submitted to AVISPA to verify its security against various security attacks. Our code consists of 4 agents: $OBU$, $BM$, $CPU$ and $BC$, two basic roles: \text{role}_{OBU}$ played by the agent $OBU$ and \text{role}_{BM}$ played by the agent $BM$, a composition role: \text{role}_{session} and \text{role}_{environment}, two additional roles: \text{role}_{BC} played by agent $BC$ and \text{role}_{CPU} played by agent $CPU$, and a part for specifies the objectives of the process. The simulation of the authentication process on AVISPA is presented in 5. The specification of roles BC and CPU are presented in Fig. 4. The specification of roles sessions and the environment with security goals are presented in Fig. 5. The specification of roles $BM_j$ and $OBU_i$ are presented in Fig. 6.

C. Informal security analysis

1) Confidentiality: Firstly, during the initialization phase, the ECDLP problem state, which even when the public key $PK_{TA} = SK_{TA} \bullet P$ of TA is given, it is really difficult and nearly impossible for attackers to deduce the secret key of TA. Likewise, $SK_{BM_j}$ and $SK_{OBU_i}$ are also difficult to calculate. Hence, the confidentiality of exchanged messages between nodes is preserved. Secondly, private personal and sensitive information is always encrypted when transmitted.
during the entire execution of our EASBF scheme. In the registration phase, both the identities of the RSUs and OBUs are always encrypted using $PK_{TA}$, the public key of TA. Except TA who has the corresponding private key, which can decrypt these messages. About the message $CPK_{OBU}^0(i, To_1, Hf(SK_{TA} || ID_{OBU}), TW_{TA})$, returned by the TA, since OBUs / RSUs need to be registered only once they can obtain it off-line for higher security. Unless someone deduces their private keys. During the authentication phase, the encrypted message $CPK_{BM}^j(AuthTok_{OBU}^i, rn_1, ID_{OBU}, TW_{OBU})$ can be decrypted to obtain the information only by the BM that having the secret key $SK_{BM}$. In addition, only persons authorized by can access the public register such as BM, AM, CA and TA. In the certificate update phase, when $Acc_{req} = <CPK_{BM}^j(ID_{OBU}, TW_{OBU}), h = Hf(ID_{OBU}, TW_{OBU}), TW_{OBU}>$, he’s only able to obtain the window of time $TW_{OBU}$ value and $h$ value. Consequently, the proposed EASBF scheme guarantees the confidentiality of important information throughout its phases.

2) Integrity: Integrity is the assurance that no adversary can alter the transmitted messages and if someone succeeds in doing so the system should detect and discover it. Firstly, based on the CDHP (Diffie-Hellman’s Computational Problem), it is really hard for adversaries to deduce the corresponding $SK_x$. Secondly, in each time, a message is exchanged, the nodes perform an integrity test based on the one way hash function. In the registration phase, both identities of BMs and OBUs are always encrypted using $PK_{TA}$ accompanied by a hash $h$. An adversary does not have the corresponding $SK_{TA}$ private key and cannot reverse the result of $h$, so he will not be able to read or falsify the messages exchanged. When the message $CPK_{OBU}^0(i, To_1, Hf(SK_{TA} || ID_{OBU}), TW_{TA})$ is returned by TA, the OBUs registered can obtain it off-line, in order to ensure that no adversary has the possibility of altering it. During the authentication phase, if the cipher text $CPK_{BM}^j(AuthTok_{OBU}^i, rn_1, ID_{OBU}, TW_{OBU})$ is altered, the BM will not be able to form the correct token $AuthTok_{OBU}^*$, therefore the validation of the integrity will fail. This is how the falsified messages are detected. In the fifth phase, certificate update; when $Acc_{req} = <CPK_{BM}^j(ID_{OBU}, TW_{OBU}), h = Hf(ID_{OBU}), TW_{OBU}>$, the system should detect and discover the falsified message.
3) **Authenticity:** Regarding the authentication phase, it is guaranteed that all messages are generated by legitimate users with a certificate. The $BM_j$ identifies the $OBU_i$ via an $AuthTok_{OBU_i}$ token constructed via the certificate extracted from the public ledger corresponding to the $ID_{OBU_i}$ identity of the $OBU_i$, and the $OBU_i$ identifies the $BM_j$ via an $AuthTok_{BM_j}$ token calculated as follows: 

$$AuthTok_{BM_j} = H_f((rn_2 \cdot PK_{BM_j}) \parallel TW_{BM_j} \parallel To_{OBU_i})$$

where this token is validated only if $rn_2 \cdot PK_{BM_j} = N_2$, because $N_2 = rn_2 \cdot P \cdot SK_{BM_j}$ can only be generated by the $BM_j$. Hence, the EASBF scheme can ensure the authentication of both $BM_j$ and $OBU_i$.

4) **Privacy and Anonymity:** To ensure anonymity, the proposed EASBF scheme should guarantee that no adversary can extract real identities when our system is in place. First, as mentioned above, our system can guarantee the confidentiality of all exchanged messages. Since the identities are always encrypted and never relayed in clear, it is impossible for adversaries to obtain any private data. In the authentication phase, vehicles instead of sending their true identities, they create $To_{OBU_i}$ a token using the one-way function and use it as their signature, and decide when it expires. It gives the responsibility to preserve the privacy to the users, which is an advantage. When a vehicle needs to update its signature, all it has to do is run again the authentication phase. Besides, real identities are entered in the public register, but only entities authorized can access them. Therefore it is difficult for an adversary to obtain the true identity of vehicle, and the signature of vehicles can be changed at any time. Hence, the proposed EASBF scheme guarantees the privacy and anonymity of vehicles.

5) **Traceability and Non-repudiation:** Based on the blockchain technology, when someone finds out that a vehicle is behaving badly, reports it to the TA, and the TA will check the public register to find out its real identity and revoke its public key. Thus, our system guarantee traceability. Meanwhile, the illegal vehicle’s true identity is revealed and marked in the public registry as a threat, which means that it cannot deny its bad behavior. As a result, the objective of non-repudiation is achieved.

6) **Non-Interactivity:** When a vehicle requests access to the fog service, it sends only one token, which is the authentication request in the case of the authentication phase or the request to access the fog service to a BM and the request for updating its certificate to a CA in the case of certificate update phase and does not need to transmit additional messages, which means no token depends on another. Hence, the proposed EASBF scheme is not interactive.

7) **Perfect Forward Secrecy (PFS):** During every session, the EASBF scheme requires participants to generate a new session key $SK_{ij}$, so that compromising a unique session key does not affect any data from other sessions. Besides, even if a component secret key is compromised, it does not affect
TABLE II: Comparison of security features between EASBF and related authentication schemes

<table>
<thead>
<tr>
<th></th>
<th>[35]</th>
<th>[36]</th>
<th>[32]</th>
<th>[33]</th>
<th>[29]</th>
<th>Our scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security goals</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Confidentiality</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Integrity</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Mutual authentication</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Privacy and anonymity</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Traceability and Non-repudiation</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Secure transmission</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Key exchange</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Non-interactivity</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Perfect Forward Secrecy</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Resistance to</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DDoS attacks</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Replay attacks</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Man-in-the-middle attacks</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Identity theft attacks</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Traffic analysis attacks</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Masquerading attacks</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Session key disclosure</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Security analysis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Formal: Random Oracle Model</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Formal: AVISPA tool</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Informal analysis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

the security of previous sessions, since the scheme utilizes the time window $TW_x$, for fresher session verification.

D. Resistance Against Potential Attacks

1) Resistance against DDoS attacks: Since several dependent nodes and Blockchain technology with Fog computing are considered in a distributed system, so even if one node goes down, the remaining nodes won’t be affected.

2) Resistance against replay attacks: Based on the one-way hash function $Hf()$ and the $TW_x$, time windows generated in a message between nodes, the EASBF scheme is robust against replay attacks. Each time, the nodes receive a message, they extract the time window attached to the message and compare it with the one encrypted in the message, and also apply the function $Hf()$ on the received message to compare its result with the hash value attached to messages. If the values do not match, they will discover that this message is not recent or has been modified and they will report it to the TA. Hence, the EASBF scheme can resist replay attacks.

3) Resistance against man-in-the-middle attacks: Based on the time windows $TW_x$ and the one-way hash function $Hf()$ with authentication tokens $AuthTok_{ix}$, the man-in-the-middle attack becomes harmless. When an attacker intercepts a communication, he would not be able to pass it over as a legitimate vehicle, because he could neither generate nor validate authentication tokens, and he could neither alter nor delay the messages exchanged because the hash function is used to control the integrity of messages with the freshness of messages.

4) Resistance against identity theft attacks: This attack is based on the theft of the identities of the vehicles which is impossible in the EASBF scheme since the identities of the nodes are never transmitted in clear and only the nodes TA, BM, AM, and CA have access to the public register (i.e., the list of vehicle identities). Hence, the identities of legitimate vehicles are protected from the attacker in the EASBF scheme.

5) Resistance against traffic analysis attacks: In this attack the attacker analyses the information collected after eavesdropping on the traffic exchanged on the network, he tries to collect information that could benefit his own personal interest. As long as the useful data is never exchanged in clear in our diagram, collecting useful data is impossible as long as the attacker is unable to deduce the secret keys $SK_x$.

6) Resistance against masquerading attacks: In this attack, the attacker is hidden using a valid identity (called a mask), and he is seeking to generate false messages. The EASBF scheme can resist against masquerading attacks since if the attacker succeeds in acquiring the identity of a legitimate vehicle, it is impossible for him to authenticate himself with the BM, because the authentication of a node does not only require the identity of the vehicle but also the signature $S$ of $Hf(SK_{TA} \parallel ID_{OBUi})$ of the TA which registered this vehicle.

7) Resistance against session key disclosure attack: In this attack, the target is the session key as discussed in our threat model in Section 111. In the EASBF scheme the attacker must obtain the random nonce $r_{n1}$, and $OBU_{i}$'s secret key $SK_{OBU_{i}}$ to generate the correct session key $SK_{ij}$. However, they cannot be obtained, since the attacker cannot break the ECDLP. Thus, EASBF is secure against the session key disclosure attack.

In Tab. II we present a comparison between EASBF and various related authentication schemes 29, 32, 33, 35, 36, regarding security features. We focused on three main characteristics: 1) Security goals; 2) Resistance to security attacks; and 3) Security analysis. For the first characteristic, we assessed the capabilities of each system to secure users and data in the IoV network. Our system guarantees confidentiality, integrity, mutual authentication, privacy and anonymity, traceability and non-repudiation, secure transmission, key exchange, non-interactivity, and perfect forward secrecy. While other schemes have no such insurance. For instance, the perfect forward secrecy is missing in 32, 35, 36, also the non-interactivity is not present in 29, 32, 33. For the second one, we compared each scheme’s capability to resist security attacks. The EASBF scheme is protected against DDoS attacks, replay attacks, man-in-the-middle attacks, identity theft attacks, traffic analysis attacks, masquerading attacks, and session key disclosure attacks. Compared to other schemes that
do not offer similar levels of protection. Such as the resistance in session key disclosure attacks which is absent in [35], [36], and also the resistance against DDoS attacks, which was not discussed in [29], [32], [33]. Security analysis includes both formal using Random Oracle Model, and AVISPA tool, and informal analysis, where we proved the validity of our previous claims. Some works do not provide the security analysis at all [35], while others just used the informal one [32], [36], which is not sufficient to prove the validity of the statements provided in these works.

VI. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the EASBF scheme with C++ under Visual Studio 2019 using the Crypto++ library. The performance metrics used in the evaluation are: (1) the computation overhead (2) the communication overhead, and (3) the storage overhead. The detailed parameter settings are summarized in Tab. III.

TABLE III: Hardware and software settings.

<table>
<thead>
<tr>
<th>Device</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor</td>
<td>Intel (R) Core (TM) i5-3470 processor @ 3.20 GHz up to 3.60 GHz</td>
</tr>
<tr>
<td>RAM</td>
<td>8.00 GB</td>
</tr>
<tr>
<td>Operating system</td>
<td>Windows 7 Professional 64-bit version</td>
</tr>
</tbody>
</table>

TABLE IV: Computation overhead of cryptography methods.

<table>
<thead>
<tr>
<th>Cryptography methods</th>
<th>Computation overhead in milliseconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECC initialization</td>
<td>10.916 ms</td>
</tr>
<tr>
<td>Generation of public and private ECC keys</td>
<td>16.592 ms</td>
</tr>
<tr>
<td>ECC multiplicative operation</td>
<td>5.627 ms</td>
</tr>
<tr>
<td>One-way hash function</td>
<td>1.297 ms</td>
</tr>
<tr>
<td>Cryptography key derivation function</td>
<td>2.856 ms</td>
</tr>
</tbody>
</table>

A. Computation Overhead

First of all, the basic operations of the registration phase are hash functions, XOR operations, concatenation operations, ECC multiplicative operations, and comparisons. Compared to other functions and operations, the XOR operation, concatenation, and comparison are negligible. For the one-way hash functions and ECC multiplicative operations, the average time measured via the C++ Chrono library is respectively 1.297 ms and 5.627 ms. In total, this phase performs 5 \( Hf() \) hash functions and 2 (●) ECC multiplicative operations. So the total time for the registration of a vehicle, \( T_{Reg} \):

\[
T_{Reg} \approx 5 \times (1.297) + 2 \times (5.627) \approx 17.739 ms
\]

Then, the basic operations of the authentication phase are the same as the registration phase plus the key derivation function. The computation time of the \( CDKf() \) key derivation functions is 2.856 ms. In total, this phase performs 5 \( Hf() \) hash functions and 6 (●) ECC multiplicative operations, and 2 \( CDKf() \) key derivation functions. Thus, the total time for the authentication of a vehicle, \( T_{Auth} \):

\[
T_{Auth} \approx 5 \times (1.297) + 6 \times (5.627) + 2 \times (2.856) \approx 45.959 ms
\]

Finally, the basic operations of the certificate update phase are the same as the authentication phase. In total, this phase performs 8 \( Hf() \) hash functions and 2 (●) ECC multiplicative operations and 2 \( CDKf() \) key derivation functions. So the total time needed to update a certificate is \( T_{Upd} \):

\[
T_{Upd} \approx 8 \times (1.297) + 2 \times (5.627) + 2 \times (2.856) \approx 27.342 ms
\]

In Tab. VI we present a comparison of the computation overhead between our scheme and various related authentication schemes [29], [32], [33], [35], [36]. The three main characteristics used for this comparison are \( T_{Reg} \) for the total time for the registration of a vehicle, \( T_{Auth} \) for the total time for the authentication of a vehicle, and \( T_{Upd} \) for the total time needed to update a certificate. Note that we used our hardware and software settings, and the computation overhead of cryptography methods listed in Tab. III and Tab. IV for the evaluation process.

As we can see in Tab. VI and Fig. 9 the EASBF scheme requires less time to authenticate a vehicle with approximately 45.959 ms compared to related schemes, namely, [32], [33], [35], [36]. Although the work in [29] requires much less time in the authentication phase compared to EASBF, but when it comes to the security features that our scheme provide in contrast, which is certainly an advantage. \( T_{Upd} \) is missing in many schemes [29], [32], [35], [36], although its present a critical characteristic for any authentication protocol. Also, note that many of the schemes have a cloud-based network architecture, where a network failure can cause all the protocol to collapse, which is not the case for our proposed scheme, since it uses a fog-based network architecture.

B. Communication Overhead

The communications overhead is based on the number of tokens transmitted between the nodes during the registration, authentication and certificate update phase. For the proposed scheme, the following tokens \( < T_{o_0}, h_0, TW_{OBU}, T_{o_1}, S, h_1, TW_{TA} > \) were transmitted between the two OBU and TA nodes during the registration phase, which gives a total of 7
tokens in this phase, denoted by $C_{\text{Cost}}^\text{Reg}$. The following tokens $<\text{AuthTok}_{OBU}, \text{rn}_1, \text{ID}_{OBU}, \text{TW}_{OBU}, \text{AuthTok}_{BMj}, \text{rn}_2, \text{TW}_{BMj}>$ were transmitted between the two OBU and BM nodes during the authentication process, which gives a total of 7 tokens in this phase, denoted by $C_{\text{Cost}}^\text{Auth}$, and the following tokens $<2\text{CTok}, 2\text{TW}_{OBU}, 2h, T_{O'}, \text{TW}_{CAj}, h_1, S>$ were transmitted between the two nodes OBU and CA during the certificate update phase. which gives a total of 10 tokens in this phase, denoted by $C_{\text{Cost}}^\text{Upd}$. So the total communications overhead for all phases is $C_{\text{Cost}}$:

$$C_{\text{Cost}} = 7 + 7 + 10 = 24 \text{ tokens}$$

In Tab. VI, we present a comparison of the communication overhead between our scheme and various related authentication schemes.

<table>
<thead>
<tr>
<th>Schemes</th>
<th>Cost(^{\text{Reg}})</th>
<th>Cost(^{\text{Auth}})</th>
<th>Cost(^{\text{Upd}})</th>
<th>Our scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>[29]</td>
<td>3</td>
<td>4</td>
<td>15</td>
<td>N/A</td>
</tr>
<tr>
<td>[35]</td>
<td>5</td>
<td>5</td>
<td>31</td>
<td>15</td>
</tr>
<tr>
<td>[36]</td>
<td>8</td>
<td>9</td>
<td>46</td>
<td>20</td>
</tr>
<tr>
<td>[32]</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>5</td>
</tr>
<tr>
<td>[33]</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
</tbody>
</table>

In Tab. VI, we present a comparison of the communication overhead between our scheme and various related authentication schemes [29], [32], [33], [35], [36]. The four main characteristics used for this comparison are $C_{\text{Cost}}^\text{Reg}$ for the total communication tokens in the registration phase, $C_{\text{Cost}}^\text{Auth}$ for the total communication tokens in the authentication phase, $C_{\text{Cost}}^\text{Upd}$ for the total communication tokens in the certificate update phase, and $C_{\text{Cost}}$ for the total communications overhead for all phases. As illustrated in Fig. 10, the EASBF scheme requires no more than 7 tokens to be exchanged in the authentication phase compared to [29], [32], [33]. Although [35], [36] has fewer tokens to be exchanged, EASBF outperforms them in the communication overload of the same phase, and will do the same in the storage overload as we will see in the next subsection, not to mention the security features it offers compared to them.

C. Storage overhead

In this sub-section, the storage cost for the vehicle node is evaluated. Fig. 11 provides a storage cost comparison between EASBF and related authentication schemes during the registration and authentication phases. To determine the storage overhead costs associated with every schemes, we have defined the following assumptions:

- Elliptic points are of 40 bytes.
- Hash digest is of 32 bytes.
- Identity and random number is of 10 bytes.
- Timestamp is of 8 bytes.
- Cryptographic key derivation function is of 64 bytes.
- Digital certificate is of 128 bytes.

It can be observed that in EASBF, the vehicle node is required to store 122 bytes at the end of the registration phase, since it has to store $\text{ID}_{OBU}$ (10 bytes), $\text{PK}_{OBU}$ (40 bytes), $\text{SK}_{OBU}$ (40 bytes) and $S$ (32 bytes). While at the end of the mutual authentication and exchanging keys phase, the vehicle node has to store 64 bytes as the output of the cryptographic key derivation function that derives the session secret key $SK_{ij}$, to provide a safe and secure communication without having to re-authenticate for future sessions. The proposed EASBF scheme demonstrated cost-effectiveness in terms of storage costs and security features compared to related schemes.

D. Computational, communication, and storage overhead requirements for the IoV applications

As a hierarchical organization, fog computing-enabled internet of vehicles requires a variety of computing services at different tiers. This means that in terms of computational, communication, and storage overheads, the performance requirements for the computing service vary. The smart vehicles perform a significant service as a critical data generator in fog computing-enabled internet of vehicles terms of sensing (e.g., cameras, radar and GPS), computational, communication and real-time storage requirements. The estimated data collected by the different sensors in a smart vehicle is approximately 25 GB/h in one day (e.g., 50 kb/s for GPS, 10 kb/s for radar, 20-60 MB/s for cameras), as discussed by Huang et al. [49]. Regarding the energy cost, the Tesla Model
S with an 85 kWh battery can travel up to 335 miles with a standard single charge [50]. However, the response times in fog computing-enabled internet of vehicles are estimated in milliseconds to subsecond, seconds to minutes, and minutes to days, at fog nodes closest to intelligent vehicles, fog aggregation nodes, and cloud servers, respectively. Hence, compared to conventional vehicular cloud computing architecture, the use of fog nodes located in proximity to smart vehicles can significantly reduce the response time by pre-processing the collected data prior to uploading and storing it in cloud servers. Through the above performance evaluation findings, we can obviously observe that our proposed EASBF scheme is efficient in terms of computation, communication, and storage overheads and acceptable for providing security in IoV applications.

VII. CONCLUSION

In this paper, we proposed an efficient authentication scheme over blockchain, named EASBF, for secure fog computing-enabled internet of vehicles. Specifically, the proposed EASBF scheme consists of five phases: initialization, registration, mutual authentication, and exchanging keys, consensus, and certificate update. Based on the elliptic curve cryptography, one-way hash function, and blockchain technology, the EASBF scheme can achieve confidentiality, integrity, authenticity, privacy, anonymity, non-repudiation, and perfect forward secrecy. Therefore, the EASBF scheme is robust against DDoS attack, replay attack, man-in-the-middle attacks, identity theft attack, traffic analysis attack, masquerading attack, and session key disclosure attack. Besides, we carried out a comprehensive security analysis, both formal and informal, using the Oracle random model and the AVISPA tool. In addition, we provided a detailed comparison of the proposed scheme with the existing state-of-the-art schemes, which have demonstrated the effectiveness of the EASBF scheme in terms of computation, communication, and storage overheads.

REFERENCES


