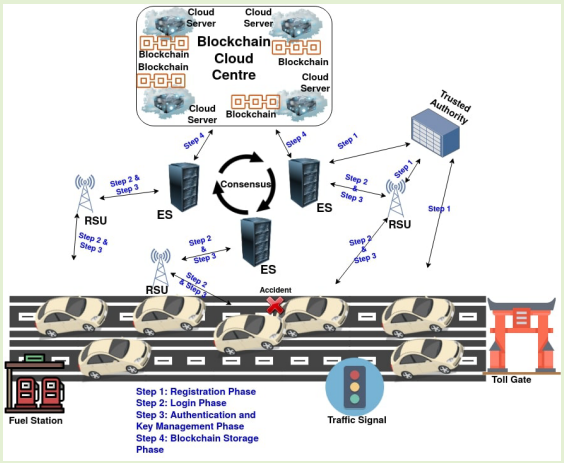
Blockchain-Enabled Certificate-Based Authentication for Vehicle Accident Detection and Notification in Intelligent Transportation Systems

Anusha Vangala, Basudeb Bera, Sourav Saha, *Student Member IEEE*, Ashok Kumar Das, *Senior Member, IEEE*, Neeraj Kumar, *Senior Member, IEEE*, and Youngho Park, *Member, IEEE*

***Abstract*—As the communications among the vehicles, the Road-Side Units (*RSU*) and the Edge Servers (*ES*) take place via wireless communication and the Internet, an adversary may take the opportunity to tamper with the data communicated among various entities in an Internet of Vehicles (IoV) environment. Therefore, it demands secure communication among the involved entities in an IoV-based Intelligent Transportation System (ITS) deployment. In this work, we design a new blockchain-enabled certificate-based authentication scheme for vehicle accident detection and notification in ITS, called BCAS-VADN. In BCAS-VADN, through the authentication process, each vehicle can securely notify accident related transactions to its nearby Cluster Head (*CH*), if an accident is detected on roads either by its own or neighbor vehicle(s). The *CH* then securely sends the transactions received from the vehicles to its *RSU* and subsequently, these transactions are also received securely by the *ES*s. The *ES* is responsible for preparing partial block containing transactions and Merkle tree root, and a digital signature on those information, and then forwarding to its associated Cloud Server (*CS*) in the Blockchain Center (*BC*) for complete block creation, verification and addition of the block using the designed consensus process. Due to blockchain technology usage, it is shown that BCAS-VADN is not only secure against various potential attacks, but also maintains transparency, immutability and decentralization of the information. Furthermore, a comprehensive comparative analysis reveals that BCAS-VADN achieves better security and more functionality attributes, and has low communication and computational overheads as compared to other competitive authentication schemes in IoV. In addition, the practical demonstration using the blockchain technology has been also provided.**



***Index Terms*—Intelligent transportation systems, authentication, vehicle accident detection, vehicle accident notification, blockchain, security.**

I. Introduction

IN PAST two decades, there has been a surge in the production of vehicles leading to the worsening of traffic on roads. Even though the sales in new vehicles has plunged down in the recent years, the amount of traffic on the roads has not declined. This has lead to increased probability of accidents on the road, thus turning an unavoidable routine activity such as daily commute into a risk. The physical and mental condition of the driver, weather condition, road condition, traffic condition and vehicle condition are the major reasons for most accidents on road with distracted driving being the topmost reason. Based on the report released by the World Health Organization (WHO) [1], "over 3,700 people die on the world's roads every day and also tens of millions of people are injured or disabled every year". It was mentioned that the most vulnerable to road users include pedestrians, children, cyclists and older people. For this purpose, WHO works with several partners including governmental and non-governmental organzations around the world for raising the profile of stop-ability of road traffic injuries and promoting good habits

Manuscript received April 28, 2020; revised June 16, 2020 and July 7, 2020; accepted July 10, 2020. Date of publication July 15, 2020; date of current version July 14, 2021. This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant 2020R1I1A3058605, in part by the Ripple Centre of Excellence Scheme, CoE in Blockchain (Sanction No. IIIT/R&D Office/Internal Projects/001/2019), IIIT Hyderabad, India, and in part by the Mathematical Research Impact Centric Support (MATRICS) project funded by the Science and Engineering Research Board (SERB), India, under Grant MTR/2019/000699. The associate editor coordinating the review of this article and approving it for publication was Dr. Alireza Jolfaei. *(Corresponding author: Neeraj Kumar.)*

Anusha Vangala, Basudeb Bera, Sourav Saha, and Ashok Kumar Das are with the Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad, Hyderabad 500 032, India (e-mail: anusha.vangala@research.iiit.ac.in; basudeb.bera@research.iiit.ac.in; sourav.saha@research.iiit.ac.in; ashok.das@iiit.ac.in, iitkgp.akdas@gmail.com).

Neeraj Kumar is with the Department of Computer Science and Engineering, Thapar University, Patiala 147 004, India, and also with the Department of Computer Science and Information Engineering, Asia University, Taichung 41354, Taiwan (e-mail: neeraj.kumar@thapar.edu).

Youngho Park is with the School of Electronics Engineering, Kyungpook National University, Daegu 41566, South Korea (e-mail: parkyh@knu.ac.kr).

Digital Object Identifier 10.1109/JSEN.2020.3009382

to address the key behaviour risk factors (i.e., drink-driving, speed, utilization of motorcycle helmets, seat-belts, and child restraints).

A comprehensive study conducted by Rolison *et al.* [2] lists the involved factors that lead to road accidents and also a classification of the factors from the most frequent to the least frequent. In addition, they provided a statistical analysis on how the driver age and gender may effect these factors.

A project, called "Communication for eSafety (COMeSafety)" [3], was developed to systematize the standards required for "vehicle-to-vehicle" and "vehicle-to-infrastructure" transmissions in an ITS environment on the European roads. One of its primary goals was to develop the European set of standards that can support wide deployment as well as implementation of co-operative ITS. The OnStar [4] was an initiative by the General Motors to activate an emergency call with the key information about the accidents.

Fogue *et al.* [5] created a system, called e-NOTIFY, that collects information from the sensors installed in a vehicle, determines the severity of an accident as *minor*, *moderate* or *severe* according to the condition that if the vehicle can be driven cautiously or not, and then sends a message to dispatch the required resources according to the severity of the accident. Furthermore, based on passenger injuries details, the severity of the passenger injury in accidents is classified into three categories as: a) "no injury", b) "non-incapacitating injury", and c) "incapacitating or fatal injury". Their designed system aims to reduce the amount of time needed to assist aid to the victims of an accident, which deems to be very crucial to save more lives.

Edge computing is a technology that enables the processing of the dense data collected from the smart devices in an Internet of Things (IoT) environment is processed at a layer before reaching at the data center. The purpose of edge computing is to reduce the communication and processing latency in the data-heavy environment, thereby boosting the performance of application. Since a part of the processing is executed at the edge layer, the load is also reduced on the cloud data center. It provides major economic and computational benefits. It also increases the privacy of the data and allows operations to be performed even with network disruptions. It adds to scalability, efficiency and resource conservation. Security in edge computing requires anonymous protection along with authenticated key exchange and access control. The de-centralization of processing is achieved by separating the network functions to decide which functions can be optimally executed on edge devices and separating the solutions into hardware and software oriented [6], [7].

As the computational power of devices improves, concerns about transmitting private information also increase. This can be solved by storing data locally and pushing network computation to the edge. The edge computing allows the computational elements to be moved to the edge devices while keeping the raw user data stored in the local devices to preserve privacy. "Federated learning", also referred as "collaborative learning", runs an algorithm over multiple edge servers or devices such that each of the servers applies the algorithm to its own data; thereby, eliminating the need for sharing local data with other servers. Thus, the federated learning enables predictive features on devices, such as smartphones with a good quality user experience while preserving secrecy of private information [8]. Such a system would require updation of the learning model deployed on the devices. Pokhrel and Choi [9] presented a mathematical framework based on renewal reward methodology for updating the learning models of the vehicle using blockchain for distribution and verification of the updated model. Their system promotes the optimal rate at which new blocks arrive, based on the delays in consensus and communication. In the "Google's Federated Learning (GFL)" [10], [11], the learning model on each vehicle sends its local updates to a global server which then calculates the global updates. The "Blockchain-based Federated Learning (BFL)" eliminates the need for the global server by allowing each vehicle to perform the global update locally. A vehicle sends its local update to a miner which verifies it and generates a block after executing the "Proof of Work (PoW)" consensus algorithm. This block is then added to the ledger which is later used by all other vehicles. Lu *et al.* [12] proposed a hybrid blockchain-based architecture that allows the vehicles to share data for asynchronous collaborative learning that uses the "Deep Reinforcement Learning (DRL)" to select nodes, and it also ensures reliability and security of model parameters for improved vehicle participation. Lu *et al.* [13] also models the data sharing between vehicles as a machine learning problem based on privacy-preserving federated learning using a blockchain-based multi-party architecture and achieves high accuracy, efficiency and security.

In this paper, through designing of a new authentication mechanism we provide vehicle accident detection and notification in an ITS environment, called BCAS-VADN. Using the blockchain technology, the transactions related to vehicle accident detection and notification are stored in a peer to peer cloud servers network maintained by the blockchain center. Due to blockchain technology, once the transactions are added in the blockchain center, these can not be tampered (modified or deleted), and immutability, transparency and decentralization properties are achieved at the same time. In addition, BCAS-VADN also provides the mechanism of end-to-end encryption, and allows to store cryptographic hash fingerprint of the transactions containing in the blocks in the blockchain while supporting a mechanism of verifying transactions integrity.

A. System Models

In this section, the following models are discussed which are essential for describing the proposed scheme.

1) Network Model: The structure of the network model is shown in Fig. 1. The architecture of the Blockchain-Enabled Intelligent Transportation System (Blockchain-ITS) consists of n_v vehicles, V_i $(i = 1, 2, \cdots, n_v)$, n_{rsu} road-side units (RSUs), RSU_l $(l = 1, 2, \cdots, n_{rsu})$, n_{es} edge servers, ES_m $(m = 1, 2, \cdots, n_{es})$ and n_{cs} cloud servers, CS_w $(w = 1, 2, \cdots, n_{cs})$. We have an Edge computing layer as in [14]. Each V_i has its on-board units (OBU_i). V_i is also equipped with several Internet of Things (IoT)-enabled in-vehicle smart

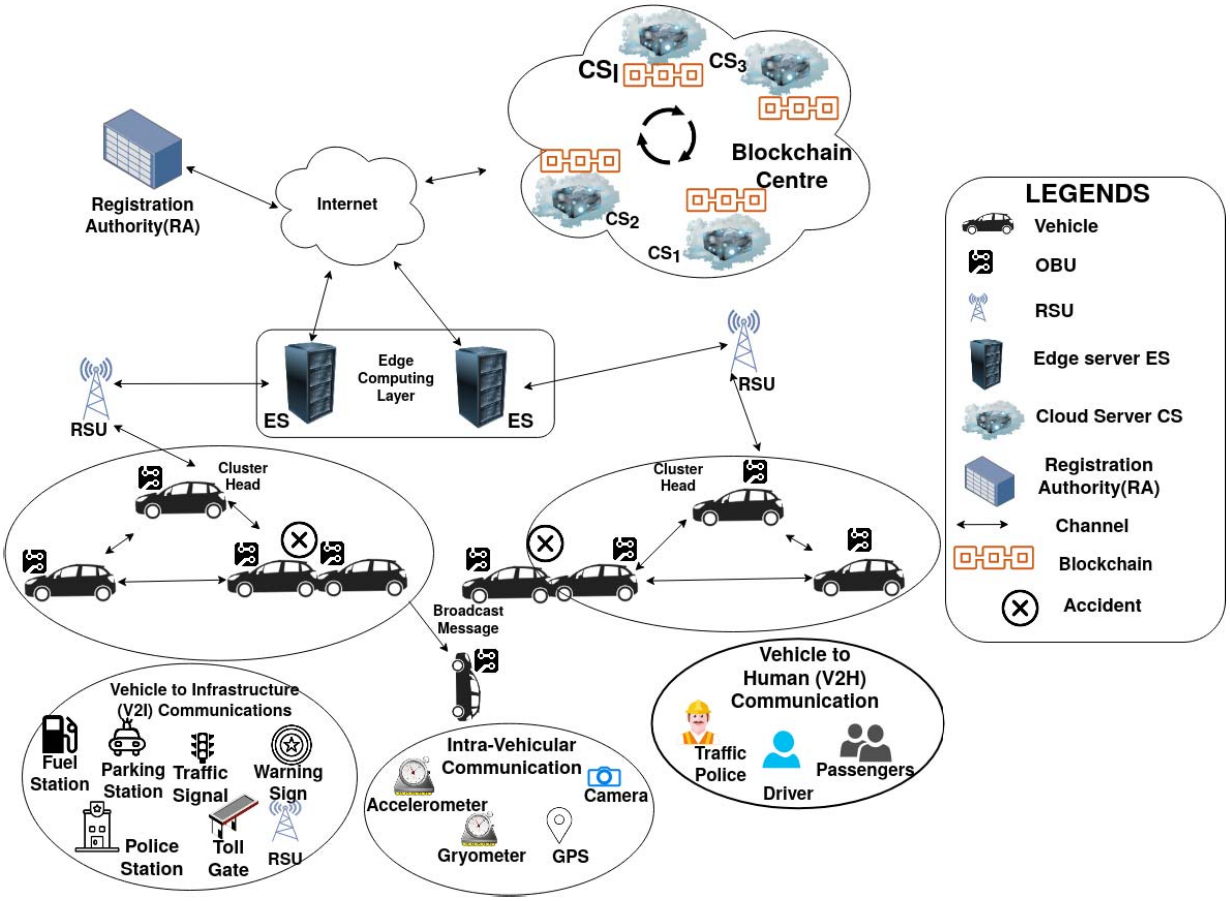


Fig. 1. Blockchain-enabled edge computing based Intelligent Transportation System (ITS).

sensors. The in-vehicle sensors helps to detect accidents and supply information about its causes. The data acquisition unit (DAU) periodically gathers data from various in-vehicle sensors and then provides the accumulated data to the OBU_i which determines if an accident happened or not, and if it is so, OBU_i notifies the situation to its nearby cluster head (CH). The CH connects via a public channel to its nearby RSU, say RSU_l. Each RSU is associated with an edge server ES_m. All the edge servers are connected to the cloud servers in the Blockchain Center (BC) via a public channel. The group of the cloud servers forms a peer to peer CS network. The cloud servers are responsible for verifying and adding the blocks containing the vehicle accident related information to the BC after executing a blockchain-based consensus algorithm.

2) Threat Model: In this threat model, we contemplate the significantly used "Dolev-Yao threat model (known as the DY model)" [15] in the networking environment. Under the DY model, an adversary $\mathcal{A}$ has the ability not only to intercept the communication messages between any two participant (i.e., V_i and CH, CH and RSU_l, and RSU_l to ES_m), but can also inject the malicious messages in the communication channel apart from altering and deleting the contents in the transmitted messages. We also consider "Canetti and Krawczyk's model (CK-adversary model)" [16], which is more powerful model as compared to the DY model. In the CK-adversary model, an adversary $\mathcal{A}$ has the capability to compromise the secret credentials, and hijacking the session keys and session states in a particular ongoing session between two parties in the network. Thus, even if a currently running session is hijacked by $\mathcal{A}$, he/she should not be able to compromise past and future session keys established between two entities. Furthermore, a vehicle's OBU may be physically captured by $\mathcal{A}$. Then, $\mathcal{A}$ can extract all the loaded information from the compromised OBU's memory by utilizing the sophisticated "power analysis attacks" [17].

B. Related Work

Liu *et al.* [18] presented a "privacy-preserving dual authenticated key agreement protocol" for secure vehicle-to-vehicle (V2V) communication in an Internet of Vehicles (IoV) environment, called as PPDAS. PPDAS relies on bilinear pairing along with trusted computing. However, PPDAS is not resilient against insider attack and it also fails to maintain the session key security under the CK-adversary model as explained in Section I-A.2.

Liu *et al.* [19] proposed an "efficient anonymous authentication scheme for the Internet of Vehicles (IoV) deployment". In their scheme, the regional management and authentication processes are performed. Furthermore, their scheme supports conditional anonymous mutual authentication, and as a result,

TABLE I
SUMMARY OF VARIOUS CRYPTOGRAPHIC TECHNIQUES USED AND LIMITATIONS OF EXISTING SCHEMES

Scheme	Year	Cryptographic Techniques	Drawbacks/Limitations
Liu *et al.* [18]	2017	* Based on privacy-preserving dual authentication & key agreement * Utilized bilinear pairings, ECC, and one-way hash function	* Not resilient against insider attack * Does not provide session key security under CK-adversary model * Expensive in computation * Does not support dynamic vehicle addition * Does not support blockchain solution
Liu *et al.* [19]	2018	* Based on anonymous authentication * Utilized bilinear pairings, ECC, one-way hash function, key derivation function (kdf) and modular exponentiation	* Expensive in computation * Does not provide session key security under CK-adversary model * Does not support dynamic vehicle addition * Does not support blockchain solution
Wu *et al.* [20]	2019	* Based on privacy protection and mutual authentication * Utilized one-way hash function and ECC	* Does not provide session key security under CK-adversary model * Does not support blockchain solution
Jiang *et al.* [21]	2019	* Based on two-factor authentication * Utilized physical unclonable function (PUF), fuzzy extractor for biometric verification, public key encryption/decryption, and one-way hash function	* Does not support dynamic vehicle addition * Expensive in computation * Does not support blockchain solution
Li *et al.* [22]	2020	* Based on hierarchical revocable authentication * Utilized one-way hash function and ECC	* Expensive in communication * Does not support blockchain solution
Tan and Chung [23]	2020	* Based on certificateless authentication * Utilized consortium blockchain for V2V group key * Utilized bilinear pairings, ECC, one-way hash function, and modular exponentiation	* Expensive in communication and computation * Not resilient against insider attack * Does not provide session key security under CK-adversary model * Does not support dynamic vehicle addition

it also preserves privacy. However, their scheme requires more computational cost due to bilinear pairing operations.

Jiang *et al.* [21] designed a two-factor authentication protocol based on "physical unclonable function (PUF)" [24] and also the "fuzzy extractor for biometric verification" [25] in an IoV environment. However, their scheme does not support dynamic vehicle addition after initial deployment.

Wu *et al.* [20] also designed a "privacy protection and mutual authentication" for secure V2V communication paradigm. Their scheme is robust against identity guessing, impersonation and replay attacks. However, their scheme does not preserve session key security under the CK-adversary model.

Li *et al.* [22] proposed a "hierarchical revocable authentication protocol in Vehicle Ad Hoc Networks (VANET)". Their approach relies on both the "self-certified public keys (SCPK)" and Schnorr signatures [26]. However, their scheme requires high communication overhead.

Tan and Chung [23] designed a "certificateless authentication scheme" in which independent session key established for each vehicle is used to avoid interference. They also employed consortium blockchain for V2V group key construction. However, this scheme requires more communication and computation costs. A summary of various cryptographic techniques used and the limitations/drawbacks of previous existing schemes is also provided in Table I.

To address the above limitations highlighted in the existing schemes, we design a novel blockchain-enabled certificate-based authentication scheme for vehicle accident detection and notification in ITS, called BCAS-VADN.

C. Paper Outline

In the next section, we give briefly the main motivation behind proposing our scheme. Section III gives a detailed description of various phases related to the proposed scheme (BCAS-VADN). Section IV gives a detailed security analysis to show the robustness of BCAS-VADN against potential attacks. The simulation results for the formal security verification using automated validation tool are provided in Section V. Next, a detailed comparative study among the proposed BCAS-VADN and other relevant state-of-art authentication protocols in ITS environment is given in Section VI. The blockchain implementation of the proposed BCAS-VADN has been provided in Section VII. The paper is finally concluded in Section VIII.

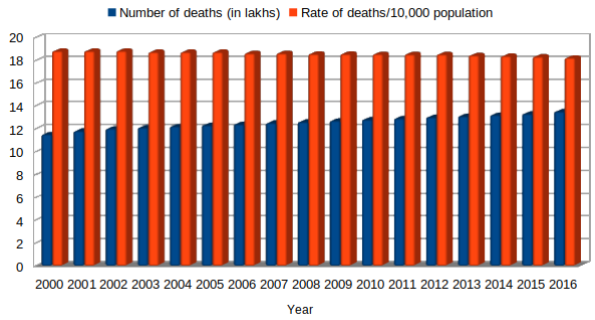


Fig. 2. Number and rate of road traffic deaths during 2000-2016 [27].

II. MOTIVATION

According to the report cited in [27], the number of fatalities caused by the road traffic accidents reached 1.35 millions in the year 2016. Fig. 2 shows the statistics on the number of deaths and the rate of deaths per 10,000 population in traffic accidents world-wide.

In particular, in Indian environment, according to the data collected by the "National Crime Records Bureau (NCRB)"

TABLE II
NOTATIONS AND THEIR MEANINGS

Symbol	Meaning
RA	Trusted registration authority
RSU_l, RID_{RSU_l}	l^{th} road-side unit and its pseudo identity
V_i, OBU_i, RID_{V_i}	i^{th} vehicle, its On-Board Unit (OBU) and pseudo identity
CS_w, RID_{CS_w}	w^{th} cloud server and its pseudo identity
ES_m, RID_{ES_m}	m^{th} edge server and its pseudo identity
$n_v, n_{rsu}, n_{es}, n_{cs}$	Number of vehicles, RSUs, edge servers and cloud servers, respectively
$Cert_X$	Certificate of an entity X created by the RA
$h(\cdot)$	A "collision-resistant cryptographic one-way hash function"
$EC(\cdot)/DC(\cdot)$	"Symmetric encryption/decryption functions", respectively
$EP(\cdot)/DP(\cdot)$	"Public key encryption/decryption functions", respectively
q	A large prime number
$GF(q)$	Galois finite field over prime q
$E_q(\alpha, \beta)$	A "non-singular elliptic curve: $y^2 = x^3 + \alpha x + \beta \pmod q$" over $GF(q)$ with $\alpha, \beta \in Z_q = \{0, 1, \cdots, q-1\}$
G	A base point G in $E_q(\alpha, \beta)$
$P + Q$	"Elliptic curve point addition" of two points $P, Q \in E_q(\alpha, \beta)$,
$k \cdot G$	"Elliptic curve point multiplication"; $k \cdot G = G + G + \cdots G$ (k times), $G \in E_q(\alpha, \beta)$, $k \in Z_q^*$
$(cpr_X, CPub_X)$	Certificate private-public key pair of an entity X
(pr_X, Pub_X)	Encryption/signature private-public key pair of an entity X
TS_x	Current system timestamp generated by an entity X
ΔT	Maximum transmission delay associated with a message
$x * y$	Modular multiplication of elements $x, y \in Z_q$
$\|\|$	Data concatenation operator
$\oplus$	Bitwise exclusive-OR operator

[28] during the years 2013-2017, 445,730 road accidents were recorded which lead to 150,093 deaths in 2017. According to the report, road accidents amount to about 85% of all the traffic accidents. Their study also highlights that 54% of the fatalities are caused due to over-speeding [28]. The statistics clearly indicate the immediate necessity for the need of a system that can shrink the number of death tolls significantly. Therefore, there is a need for identifying a vehicle collision, and storing the crash data with immunity towards data mutability in a decentralized environment.

The communications among various entities in an ITS environment take place via public channels as in other networking environments [29]–[32]. An adversary can then take opportunity to tamper with the vehicle accident related important data in between the communication among the entities and also mount various potential attacks, such as replay, impersonation, man-in-the-middle, privileged-insider, etc. To handle these issues, we aim to design a new blockchain-enabled certificate-based authentication scheme for vehicle accident detection and notification in ITS (BCAS-VADN).

III. THE PROPOSED SCHEME

This section gives a detailed description of the proposed blockchain-based scheme (BCAS-VADN) phase-wise. Various symbols and their significance tabulated in Table II are used to describe and analyze BCAS-VADN. Furthermore, a typical assumption about the clock synchronization among all the communicating entities (vehicles, RSUs, edge servers and cloud servers in the blockchain center) is also applied in the proposed BCAS-VADN as in other schemes too [33]–[37].

In the following, various phases are discussed.

A. System Initialization Phase

In this section, it is presumed that there exists a trusted registration authority (RA) who will be solely responsible for enrolling all the communicating entities in the network. The following steps are then executed by the RA to accomplish the task of all system-related parameters generation:

Step S1: RA first picks a "non-singular elliptic curve of the form $E_q(\alpha, \beta) : y^2 = x^3 + \alpha x + \beta \pmod q$" over a Galois field $GF(q)$, where q being a large prime so that the "Elliptic Curve Discrete Logarithm Problem (ECDLP)" turns out to be intractable and $\alpha, \beta \in Z_q$ are constants with the condition: $4\alpha^3 + 27\beta^2 \neq 0 \pmod q$, and also a base point $G \in E_q(\alpha, \beta)$ whose order is as big as q.

Step S2: RA picks a "collision-resistant one-way hash function, say $h(\cdot)$", symmetric encryption and decryption functions as $EC(\cdot)$ and $DC(\cdot)$ respectively, and public-key encryption and decryption functions as $EP(\cdot)$ and $DP(\cdot)$ respectively.

Step S3: RA then selects its own private and public key pair as $(pr_{RA} \in Z_q^*, Pub_{RA})$ where $Pub_{RA} = pr_{RA} \cdot G$. In addition, the RA does the following: 1) for each cloud server CS_w, $(w = 1, 2, \cdots, n_{cs})$, RA picks their distinct certificate private and public key pair as $(cpr_{CS_w} \in Z_q^*, CPub_{CS_w} = cpr_{CS_w} \cdot G)$ 2) for each edge server ES_m, $(m = 1, 2, \cdots, n_{es})$, RA picks their distinct certificate private and public key pair as $(cpr_{ES_m} \in Z_q^*, CPub_{ES_m} = cpr_{ES_m} \cdot G)$, 3) for each RSU_l, $(l = 1, 2, \cdots, n_{rsu})$, RA picks their unique certificate private and public key pair as $(cpr_{RSU_l} \in Z_q^*, CPub_{RSU_l} = cpr_{RSU_l} \cdot G)$, and 4) for each vehicle V_i, $(i = 1, 2, \cdots, n_v)$, RA picks their unique certificate private and public key pair as $(cpr_{V_i} \in Z_q^*, CPub_{V_i} = cpr_{V_i} \cdot G)$

Step S4: Finally, the RA keeps pr_{RA}, $\{cpr_{CS_w} | w = 1, 2, \cdots, n_{cs}\}$, $\{cpr_{ES_m} | m = 1, 2, \cdots, n_{es}\}$, $\{cpr_{RSU_l} | l = 1, 2, \cdots, n_{rsu}\}$ and $\{cpr_{V_i} | i = 1, 2, \cdots, n_v\}$ as private and declares Pub_{RA}, $\{CPub_{CS_w} | w = 1, 2, \cdots, n_{cs}\}$, $\{CPub_{ES_m} | m = 1, 2, \cdots, n_{es}\}$, $\{CPub_{RSU_l} | l = 1, 2, \cdots, n_{rsu}\}$ and $\{CPub_{V_i} | i = 1, 2, \cdots, n_v\}$, $h(\cdot)$, $EC(\cdot)/DC(\cdot)$ and $EP(\cdot)/DP(\cdot)$ as public.

B. Enrollment Phase

Under this phase, the RA will perform registration of all the entities that will be deployed in the network.

1) Cloud Server Enrollment: For cloud servers enrollment, the following steps are mandatory:

Step ECS1: For each deployed cloud server CS_w, $(w = 1, 2, \cdots, n_{cs})$, RA generates a unique identity ID_{CS_w}, its associated pseudo-identity $RID_{CS_w} = h(ID_{CS_w} || cpr_{CS_w} || RTS_{CS_w})$, where RTS_{CS_w} corresponds to the registration time, and certificate $Cert_{CS_w} = pr_{RA} + h(RID_{CS_w} || CPub_{CS_w} || Pub_{RA}) * cpr_{CS_w} \pmod q$, and then sends the information $\langle RID_{CS_w}, Cert_{CS_w} \rangle$ to the respective CS_w via secure channel.

Step ECS2: Once the information $\langle RID_{CS_w}, Cert_{CS_w} \rangle$ are received by CS_w, it generates its own encryption/signature private-public key pair by picking a random secret $pr_{CS_w} \in Z_q^*$ as private key and then calculating its public key $Pub_{CS_w} = pr_{CS_w} \cdot G$. Finally, CS_w stores the credentials $\{RID_{CS_w}, Cert_{CS_w}, pr_{CS_w}\}$ in its secure database and makes Pub_{CS_w} as public. RA also deletes all the private keys cpr_{CS_w} of CS_w for security reasons.

2) Edge Server Enrollment: In a similar way, for edge servers enrollment, the following steps are also mandatory as executed in Section III-B.1:

Step EES1: For each deployed edge server ES_m, ($m = 1, 2, \cdots, n_{es}$), RA creates a unique identity ID_{ES_m}, its associated pseudo-identity $RID_{ES_m} = h(ID_{ES_m} \,||cpr_{ES_m} \,||RTS_{ES_m})$ where RTS_{ES_m} is the registration time, a certificate $Cert_{ES_m} = pr_{RA} + h(RID_{ES_m} \,||CPub_{ES_m} \,||Pub_{RA}) * cpr_{ES_m} \pmod q$, and the pairwise pre-shared secret keys between each ES_m and its associated RSU, say RSU_l as K_{RSU_l,ES_m}. After that the RA sends the credentials $\{RID_{ES_m}, Cert_{ES_m}, K_{RSU_l,ES_m}\}$ securely to the corresponding ES_m.

Step EES2: After the credentials $\langle RID_{ES_m}, Cert_{ES_m}, K_{RSU_l,ES_m}\rangle$ are received by ES_m, it starts creating its own encryption/signature private-public key pair (pr_{ES_m}, Pub_{ES_m}), where $pr_{ES_m} \in Z_q^*$ is a random secret and $Pub_{ES_m} = pr_{ES_m} \cdot G$. ES_m proceeds to store the registration information $\langle RID_{ES_m}, Cert_{ES_m}, pr_{ES_m}, K_{RSU_l,ES_m}\rangle$ in its secure database and declares Pub_{ES_m} as public. In addition, RA also erases all the private keys cpr_{ES_m} of ES_m for security reasons.

3) RSU Enrollment: The RA does the following steps to complete the registration task of each RSU to be deployed in the network:

Step ERSU1: For each deployed RSU, RSU_l, ($l = 1, 2, \cdots, n_{rsu}$), RA chooses a unique identity ID_{RSU_l}, its corresponding pseudo-identity $RID_{RSU_l} = h(ID_{RSU_l} \,||cpr_{RSU_l} \,||RTS_{RSU_l})$, where RTS_{RSU_l} denotes the registration time, and certificate $Cert_{RSU_l} = pr_{RA} + h(RID_{RSU_l} \,||CPub_{RSU_l} \,||Pub_{RA}) * cpr_{RSU_l} \pmod q$, and then sends $\langle RID_{RSU_l}, Cert_{RSU_l}, K_{RSU_l,ES_m}\rangle$ to RSU_l via secure channel.

Step ERSU2: Upon receiving the registration information from the RA, RSU_l generates a random private key $pr_{RSU_l} \in Z_q^*$ and computes the respective public key as $Pub_{RSU_l} = pr_{RSU_l} \cdot G$. RSU_l declares Pub_{RSU_l} as public and then stores the credentials $\{RID_{RSU_l}, Cert_{RSU_l}, K_{RSU_l,ES_m}, pr_{RSU_l}\}$ in its secure database. RA also discards all the private keys cpr_{RSU_l} of RSU_l for security reasons.

4) Vehicle Enrollment: For registering a vehicle V_i, ($i = 1, 2, \cdots, n_v$), the RA also follows the following steps:

Step EV1: For each registered V_i, RA first requires to pick unique identity ID_{V_i}. Next, RA calculates the pseudo-identity $RID_{V_i} = h(ID_{V_i} \,||cpr_{V_i} \,||RTS_{V_i})$, where RTS_{V_i} signifies the registration time when V_i was registered in the system and certificate $Cert_{V_i} = pr_{RA} + h(RID_{V_i} \,||CPub_{V_i} \,||Pub_{RA}) * cpr_{V_i} \pmod q$.

Step EV2: RA also selects a random $rs_{V_i} \in Z_q^*$, and computes the partial private key $ppr_{V_i} = h(pr_{RA} \,||cpr_{V_i} \,||rs_{V_i})$ and corresponding public key $PK_{V_i} = ppr_{V_i} \cdot G$ for each V_i, and stores the registration information $\{RID_{V_i}, Cert_{V_i}, ppr_{V_i}\}$ into the tamper-resistant on-board unit device OBU_i of the vehicle V_i. Finally, RA deletes all the private keys cpr_{V_i} for security reasons and makes PK_{V_i} as public.

All the registration relations information stored in various entities are briefed in Fig. 3.

Vehicle (V_i)
RID_{V_i}, $Cert_{V_i}$, ppr_{V_i}
RSU (RSU_l)
RID_{RSU_l}, $Cert_{RSU_l}$, K_{RSU_l,ES_m}, pr_{RSU_l}
Edge Server (ES_m)
RID_{ES_m}, $Cert_{ES_m}$, pr_{ES_m}, K_{RSU_l,ES_m}
Cloud Server (CS_w)
RID_{CS_w}, $Cert_{CS_w}$, pr_{CS_w}

Fig. 3. Credentials stored in registered entities.

C. Authentication Phase

In this section, we mainly discuss authentication and key agreement process in two cases: 1) between a vehicle and its associated cluster head (V2CH) and 2) between a cluster head and its RSU (CH2RSU). At the end of this process, the communicating entities will be able to establish session keys among them after their mutual authentication. Once the session keys are established successfully, those keys are utilized for secure communication among them for the accident related detection and notification information in form of the transactions.

1) V2CH Authentication: Since the vehicles movement is dynamic in nature, we consider a dynamic clustering as suggested by Kakkasageri and Manvi [38]. In their approach, various cluster members are picked based on various factors, such as "relative speed of vehicles" and "direction for dynamic clustering". A cluster head (CH) is selected among the available members in a cluster based on various metrics: a) "connectivity degree", b) "average speed" and c) "time to leave the road intersection". They mentioned that it is possible for the cluster members to re-connect with some cluster head(s) with similar mobility pattern which are passing through an intersection of a lane, for instance. They demonstrated that it is a quite possible task to form clusters efficiently. Jolfaei *et al.* [14], [39] proposed a three-layered architecture for ITS with the following layers: a) the bottom layer consisting of all the edge devices and IoT sensors, b) the middle layer consisting of RSUs which stores data temporarily and applies analytics, and c) the top layer with servers is to process and store data permanently. The devices are grouped as belonging to an RSU zone with a group leader that communicates with the associated RSU. Ezabadi *et al.* [39] also proposed a hierarchical architecture in which vehicles are grouped into a vehicle group, which are further grouped into zones and regions. As in Dua *et al.* [33], we utilize the dynamic clustering in this work.

Assume that a vehicle (V_i) wants to make secure communication with its CH of a cluster. To achieve this goal, the following steps need to be executed by both V_i and CH:

Step AVCH1: V_i (OBU_i) as the initiator generates a random secret $r_{V_i} \in Z_q^*$ and current timestamp TS_V, and calculates $X_{V_i} = h(RID_{V_i} \,||ppr_{V_i} \,||r_{V_i} \,||TS_V) \cdot G$ and the signature on r_{V_i} as $Sig_{r_{V_i}} = h(RID_{V_i} \,||ppr_{V_i} \,||r_{V_i} \,||TS_V) + h(RID_{V_i} \,||Cert_{v_i} \,||TS_V) * ppr_{V_i} \pmod q$. After these calculations, V_i dispatches the request message $Msg_{VCH1} = \langle RID_{V_i}, Cert_{V_i}, X_{V_i}, Sig_{r_{V_i}}, TS_V\rangle$ to its cluster head CH via open channel.

Step AVCH2: After receiving the message Msg_{VCH1} at time TS_V^*, CH (OBU_{CH}) checks validity of the timeliness of TS_V

by $|TS_V^* - TS_V| < \Delta T$. If the condition is verified, CH further verifies the certificate $Cert_{V_i}$ of V_i by the criteria: $Cert_{V_i} \cdot G = Pub_{RA} + h(RID_{V_i} \| CPub_{V_i} \| Pub_{RA}) \cdot CPub_{V_i}$. If it is valid, CH also verifies the attached signature $Sig_{r_{V_i}}$ with the condition: $Sig_{r_{V_i}} \cdot G = X_{V_i} + h(RID_{V_i} \| Cert_{v_i} \| TS_V) \cdot PK_{V_i}$. If this condition is true, V_i is authenticated by CH.

Step AVCH3: CH then generates a random secret $r_{CH1} \in Z_q^*$ and current timestamp TS_{CH1}, and computes $Y_{CH} = h(RID_{CH} \| ppr_{CH} \| r_{CH1} \| TS_{CH1}) \cdot G$, the Diffie-Hellman type secret key $DHK_{CH,V_i} = h(RID_{CH} \| ppr_{CH} \| r_{CH1} \| TS_{CH1}) \cdot X_{V_i}$, the session key shared with V_i as $SK_{CH,V_i} = h(DHK_{CH,V_i} \| Sig_{r_{V_i}} \| Cert_{V_i} \| Cert_{CH} \| TS_{CH1})$, and signature on r_{CH1} and SK_{CH,V_i} as $Sig_{CH1} = h(RID_{CH} \| ppr_{CH} \| r_{CH1} \| TS_{CH1}) + h(RID_{CH} \| RID_{V_i} \| SK_{CH,V_i} \| TS_V \| TS_{CH1}) * ppr_{CH} \pmod q$. Next, CH dispatches the response message $Msg_{VCH2} = \langle RID_{CH}, Cert_{CH}, Y_{CH}, Sig_{CH1}, TS_{CH1} \rangle$ to its neighbor V_i via open channel.

Step AVCH4: Let the message Msg_{VCH2} be received by V_i at time TS_{CH1}^*. V_i first checks the timeliness of TS_{CH1} by $|TS_{CH1}^* - TS_{CH1}| < \Delta T$ and it is true, V_i validates the certificate $Cert_{CH}$ of CH by $Cert_{CH} \cdot G = Pub_{RA} + h(RID_{CH} \| CPub_{CH} \| Pub_{RA}) \cdot CPub_{CH}$. If the certificate is valid, V_i further computes the Diffie-Hellman type secret key $DHK_{V_i,CH} = h(RID_{V_i} \| ppr_{V_i} \| r_{V_i} \| TS_V) \cdot Y_{CH}$, the session key shared with CH as $SK_{V_i,CH} = h(DHK_{V_i,CH} \| Sig_{r_{V_i}} \| Cert_{V_i} \| Cert_{CH} \| TS_{CH1})$. Next, V_i checks the signature Sig_{CH1} by $Sig_{CH1} \cdot G = Y_{CH} + h(RID_{CH} \| RID_{V_i} \| SK_{V_i,CH} \| TS_V \| TS_{CH1}) \cdot PK_{CH}$. If the signature is valid, CH is also authenticated by V_i.

After mutual authentication, both V_i and CH will share the same session key $SK_{V_i,CH}$ $(= SK_{CH,V_i})$ as $DHK_{V_i,CH} = h(RID_{V_i} \| ppr_{V_i} \| r_{V_i} \| TS_V) \cdot Y_{CH} = (h(RID_{V_i} \| ppr_{V_i} \| r_{V_i} \| TS_V) * h(RID_{CH} \| ppr_{CH} \| r_{CH1} \| TS_{CH1})) \cdot G = h(RID_{CH} \| ppr_{CH} \| r_{CH1} \| TS_{CH1}) \cdot X_{V_i} = DHK_{CH,V_i}$.

2) CH2RSU Authentication: If a cluster head (CH) wishes to establish a secure communication with its associated RSU, say RSU_l, they need to carry out the following steps:

Step ACHR1: CH generates a random secret $r_{CH_2} \in Z_q^*$ and current timestamp TS_{CH_2} to calculate $S_{CH} = h(r_{CH_2} \| ppr_{CH} \| TS_{CH_2}) \cdot G$ and the signature on r_{CH_2} as $Sign_{r_{CH_2}} = h(r_{CH_2} \| ppr_{CH} \| TS_{CH_2}) + h(RID_{CH} \| CPub_{CH} \| CPub_{RSU_l} \| Cert_{CH} \| Pub_{RA} \| TS_{CH_2}) * ppr_{CH} \pmod q$. After computing these components, CH dispatches the request message $Msg_{CHRSU_1} = \langle RID_{CH}, Cert_{CH}, S_{CH}, Sign_{r_{CH_2}}, TS_{CH_2} \rangle$ to the RSU_l via public channel.

Step ACHR2: After receiving Msg_{CHRSU_1} at time $TS_{CH_2}^*$, RSU_l checks the validity of TS_{CH_2} by $|TS_{CH_2}^* - TS_{CH_2}| < \Delta T$. If it is valid, RSU_l verifies the $Cert_{CH}$ by $Cert_{CH} \cdot G = Pub_{RA} + h(RID_{CH} \| CPub_{CH} \| Pub_{RA}) \cdot CPub_{CH}$, and if it is valid, RSU_l proceeds to validate the signature by $Sign_{r_{CH_2}} \cdot G = S_{CH} + h(RID_{CH} \| CPub_{CH} \| CPub_{RSU_l} \| Cert_{CH} \| Pub_{RA} \| TS_{CH_2}) \cdot PK_{CH}$. If the signature is valid, CH is authenticated by RSU_l.

Step ACHR3: Next, RSU_l generates a random secret $r_{RSU} \in Z_q^*$ and current timestamp TS_{RSU} for computing $T_{RSU} = h(r_{RSU} \| pr_{RSU_l} \| TS_{RSU}) \cdot G$, $DHK_{RSU,CH} = h(r_{RSU} \| pr_{RSU_l} \| TS_{RSU}) \cdot S_{CH}$, $\gamma = h(K_{RSU_l,ES_m} \| pr_{RSU_l} \| Cert_{CH} \| Cert_{RSU_l} \| TS_{RSU})$, $V_{RSU} = \gamma \oplus h(DHK_{RSU,CH} \| Sign_{r_{CH_2}} \| TS_{RSU})$ and signature on r_{RSU} and $DHK_{RSU,CH}$ as $Sign_{r_{RSU}} = h(r_{RSU} \| pr_{RSU_l} \| TS_{RSU}) + h(RID_{RSU_l} \| Cert_{RSU_l} \| DHK_{RSU,CH} \| Pub_{RA} \| TS_{RSU}) \cdot pr_{RSU_l} \pmod q$. RSU_l then sends the response message $Msg_{CHRSU_2} = \langle RID_{RSU_l}, Cert_{RSU_l}, T_{RSU}, V_{RSU}, Sign_{r_{RSU}}, TS_{RSU} \rangle$ to CH via public channel.

Step ACHR4: After receiving Msg_{CHRSU_2} at time TS_{RSU}^*, CH checks the validity of TS_{RSU} by $|TS_{RSU}^* - TS_{RSU}| < \Delta T$. If validation passes, CH verifies $Cert_{RSU_l}$ by $Cert_{RSU_l} \cdot G = Pub_{RA} + h(RID_{RSU_l} \| CPub_{RSU_l} \| Pub_{RA}) \cdot CPub_{RSU_l}$. If certificate is valid, CH further computes $DHK_{CH,RSU} = h(r_{CH_2} \| ppr_{CH} \| TS_{CH_2}) \cdot T_{RSU}$, $\gamma' = h(K_{RSU_l,ES_m} \| pr_{RSU_l} \| Cert_{CH} \| Cert_{RSU_l} \| TS_{RSU}) = V_{RSU} \oplus h(DHK_{CH,RSU} \| Sign_{r_{CH_2}} \| TS_{RSU})$, and verifies the signature $Sign_{r_{RSU}}$ by $Sign_{r_{RSU}} \cdot G = T_{RSU} + h(RID_{RSU_l} \| Cert_{RSU_l} \| DHK_{CH,RSU} \| Pub_{RA} \| TS_{RSU}) \cdot CPub_{RSU_l}$. If the signature is found to be valid, CH authenticates RSU_l. Next, CH generates a current timestamp TS_{CH_3} and computes the session key shared with RSU_l as $SK_{CH,RSU} = h(DHK_{CH,RSU} \| \gamma' \| Sign_{r_{CH_2}} \| Sign_{r_{RSU}} \| TS_{CH_3})$, and its verifier as $SKV_{CH,RSU} = h(SK_{CH,RSU} \| TS_{CH_3})$. CH dispatches the acknowledgment message $Msg_{CHRSU_3} = \langle SKV_{CH,RSU}, TS_{CH_3} \rangle$ to RSU_l via open channel.

Step ACHR5: After receiving Msg_{CHRSU_3} at time $TS_{CH_3}^*$, RSU_l checks the validity of TS_{CH_3} by $|TS_{CH_3}^* - TS_{CH_3}| < \Delta T$. If it is valid, RSU_l calculates the session key shared with CH as $SK_{RSU,CH} = h(DHK_{RSU,CH} \| \gamma \| Sign_{r_{CH_2}} \| Sign_{r_{RSU}} \| TS_{CH_3})$, and checks the session key verifier $SKV_{RSU,CH}$ by validating the relation: $SKV_{RSU,CH} = h(SK_{RSU,CH} \| TS_{CH_3})$. If it holds, both CH and RSU keep the same shared session key $SK_{CH,RSU}$ $(= SK_{RSU,CH})$ for secret communication between them.

Overall phase is finally summarized in Fig. 4.

D. Blockchain Verification and Addition Phase

In this section, we discuss how the blocks are formed and added into the blockchain center (BC). The blocks are created in two levels: 1) partially by an edge server (ES_m) and 2) full block by a cloud server (CS_w) in the BC.

Step B1: A vehicle (V_i) first creates a transaction, say Tx_i, once a vehicle accident (either the same vehicle or its nearby neighbor vehicle(s)) is detected by its own OBU_i. The format of Tx_i contains the following fields: a) time of accident takes place $(Time_{acd})$, b) location of accident (Loc_{acd}), c) ID of the vehicle in accident (ID_{Vacd}), d) ID of reporting vehicle (ID_{Vrep}), e) direction of accident vehicle (Dir_{Vacd}), f) position of accident vehicle (Pos_{Vacd}), g) level of accident vehicle $(Level_{Vacd})$ and h) severity of the passengers in the accident vehicle (Sev_{Vacd}), which can be either "no injury" or "non-incapacitating injury" or "incapacitating or fatal injury". Note that the direction of a vehicle is obtained from the angle of the horizontal tilt sensor(s), and also the position of the vehicle is obtained from the acceleration values taken as an integration of the amplitude of peak acceleration with respect to the time duration. Based on the direction and position, the accident is further classified into three levels: a) minor, b) moderate and c) severe [5].

Cluster head (CH)	Road Side Unit (RSU_l)
Generate random secret $r_{CH_2} \in Z_q^*$, current timestamp TS_{CH_2}. Calculate $S_{CH} = h(r_{CH_2}\|\|ppr_{CH}\|\|TS_{CH_2}) \cdot G$, $Sign_{r_{CH_2}} = h(r_{CH_2}\|\|ppr_{CH}\|\|TS_{CH_2}) + h(RID_{CH}\|\|CPub_{CH}\|\|CPub_{RSU_l}\|\| Cert_{CH}\|\|Pub_{RA}\|\|TS_{CH_2}) * ppr_{CH} \pmod q$. $Msg_{CHRSU_1} = \{RID_{CH}, Cert_{CH}, S_{CH}, Sign_{r_{CH_2}}, TS_{CH_2}\}$ (via public channel)	
	Check validity of TS_{CH_2}. Accept/Reject? If so, verify $Cert_{CH}$ as $Cert_{CH} \cdot G \stackrel{?}{=} Pub_{RA} + h(RID_{CH}\|\|CPub_{CH}\|\|Pub_{RA}) \cdot CPub_{CH}$ If valid, verify signature as $Sign_{r_{CH_2}} \cdot G \stackrel{?}{=} S_{CH} + h(RID_{CH}\|\|CPub_{CH}\|\|CPub_{RSU_l}\|\|Cert_{CH} \|\|Pub_{RA}\|\|TS_{CH_2}) \cdot PK_{CH}$ If valid, CH is authenticated by RSU_l. Generate random secret $r_{RSU} \in Z_q^*$, current timestamp TS_{RSU}. Compute $T_{RSU} = h(r_{RSU}\|\|pr_{RSU_l}\|\|TS_{RSU}) \cdot G$, $DHK_{RSU,CH} = h(r_{RSU}\|\|pr_{RSU_l} \|\|TS_{RSU}) \cdot S_{CH}$, $\gamma = h(K_{RSU_l,ES_m}\|\|pr_{RSU_l}\|\|Cert_{CH} \|\|Cert_{RSU_l}\|\|TS_{RSU})$, $V_{RSU} = \gamma \oplus h(DHK_{RSU,CH} \|\|Sign_{r_{CH_2}}\|\|TS_{RSU})$, signature on r_{RSU} and $DHK_{RSU,CH}$ as $Sign_{r_{RSU}} = h(r_{RSU}\|\|pr_{RSU_l} \|\|TS_{RSU}) + h(RID_{RSU_l}\|\|Cert_{RSU_l}\|\|DHK_{RSU,CH} \|\|Pub_{RA}\|\|TS_{RSU}) \cdot pr_{RSU_l} \pmod q$. $Msg_{CHRSU_2} = \{RID_{RSU_l}, Cert_{RSU_l}, T_{RSU}, V_{RSU}, Sign_{r_{RSU}}, TS_{RSU}\}$ (via public channel)
Check validity of TS_{RSU}. Accept/Reject? If so, verify if $Cert_{RSU_l} \cdot G \stackrel{?}{=} Pub_{RA} + h(RID_{RSU_l}\|\|CPub_{RSU_l}\|\|Pub_{RA}) \cdot CPub_{RSU_l}$ If valid, compute $DHK_{CH,RSU} = h(r_{CH_2}\|\| ppr_{CH}\|\|TS_{CH_2}) \cdot T_{RSU}$, $\gamma' = h(K_{RSU_l,ES_m}\|\|pr_{RSU_l} \|\|Cert_{CH}\|\|Cert_{RSU_l}\|\|TS_{RSU})$ $= V_{RSU} \oplus h(DHK_{CH,RSU}\|\|Sign_{r_{CH_2}}\|\|TS_{RSU})$. If valid, verify signature $Sign_{r_{RSU}}$ as $Sign_{r_{RSU}} \cdot G \stackrel{?}{=} T_{RSU} + h(RID_{RSU_l}\|\|Cert_{RSU_l}\|\|DHK_{CH,RSU} \|\|Pub_{RA}\|\|TS_{RSU}) \cdot CPub_{RSU_l}$. If signature is valid, RSU_l is authenticated by CH. Generate current timestamp TS_{CH_3} and compute $SK_{CH,RSU} = h(DHK_{CH,RSU}\|\|\gamma'\|\|Sign_{r_{CH_2}} \|\|Sign_{r_{RSU}}\|\|TS_{CH_3})$, session key verifier $SKV_{CH,RSU} = h(SK_{CH,RSU}\|\|TS_{CH_3})$. $Msg_{CHRSU_3} = \{SKV_{CH,RSU}, TS_{CH_3}\}$ (via public channel)	
	Check validity of TS_{CH_3}. Accept/Reject? If valid, compute $SK_{RSU,CH} = h(DHK_{RSU,CH}\|\|\gamma\|\|Sign_{r_{CH_2}} \|\|Sign_{r_{RSU}}\|\|TS_{CH_3})$. Check $SKV_{RSU,CH} \stackrel{?}{=} h(SK_{RSU,CH}\|\|TS_{CH_3})$. If so, session key is considered as valid.
Both CH and RSU_l store the shared session key $SKV_{CH,RSU} (= SKV_{RSU,CH})$	

Fig. 4. Summary of authentication between *CH* and *RSU*$_l$.

Step B2: Next, V_i will generate a signature Sig_{Tx_i} using the signature generation algorithm of the "Elliptic Curve Digital Signature Algorithm (ECDSA)" [40] on the message $msg = h(Tx_i)$ with the help of its own private key ppr_{V_i}. After that V_i will use the already established session key $SK_{V_i,CH}$ with its neighbor CH to send the notification message $Msg_{notif} = \langle RID_{V_i}, EC_{SK_{V_i,CH}}(Tx_i, RID_{V_i}), Sig_{Tx_i}\rangle$ to CH via public channel.

Step B3: After receiving Msg_{notif}, CH decrypts $EC_{SK_{V_i,CH}}(Tx_i, RID_{V_i})$ using the established session key $SK_{V_i,CH}$ shared with V_i to retrieve $(Tx_i, RID'_{V_i}) = DC_{SK_{V_i,CH}}[EC_{SK_{V_i,CH}}(Tx_i, RID_{V_i})]$ and check if $RID'_{V_i} = RID_{V_i}$. If it is valid, CH validates the signature Sig_{Tx_i} on $msg = h(Tx_i)$ using the ECDSA signature verification algorithm [40]. If the signature is valid, CH sends the information $\{Tx_i, Sig_{Tx_i}\}$ securely using the secret key $SK_{CH,RSU}$ to its associated RSU_l.

Step B4: RSU_l validates the Sig_{Tx_i} on received transaction Tx_i using ECDSA signature verification algorithm. If the signature is valid, RSU_l sends securely the information $\{Tx_i, Sig_{Tx_i}\}$ to its associated ES_m using the pre-shared key K_{RSU_l,ES_m}.

Block Header	
Merkle Tree Root	MTR_i
Owner of Block	OB_i (ES_m)
Public key of signer	Pub_{ES_m}
Block Payload (Transactions)	
List of n_t transactions and their signatures	$\{(Tx_i, Sig_{Tx_i}) \|i = 1, 2, \cdots, n_t\}$
Signature on all transactions (Tx_i)	Sig_{Block_i}

Fig. 5. Structure of a partial block *PartialBlock*$_i$.

Step B5: If the signature Sig_{Tx_i} is validated successfully on the received Tx_i, ES_m stores (Tx_i, Sig_{Tx_i}). After filtering all the transactions, assume that ES_m has a list of n_t important transactions which need to be applied in block formation. ES_m then creates a partial block, say $PartialBlock_i$, containing n_t transactions Tx_i and its signature Sig_{Tx_i}, which has the formats shown in Fig.5. The Merkle tree root (MTR_i) is obtained from the n_t transactions Tx_i $(i = 1, 2, \cdots, n_t)$. Next, $PartialBlock_i$ is forwarded to its respective cloud server CS_w.

Block Header	
Block Version	$BVer_i$
Previous Block Hash	PBH_i
Timestamp	TS_i
Merkle Tree Root	MTR_i
Owner of Block	OB_i (ES_m)
Public key of signer	Pub_{ES_m}
Block Payload (Transactions)	
List of n_t transactions and their signatures	$\{(Tx_i, Sig_{Tx_i}) \mid i = 1, 2, \cdots, n_t\}$
Signature on all transactions (Tx_i)	Sig_{Block_i}
Current Block Hash	CBH_i

Fig. 6. Structure of a full block $Block_i$.

Step B6: After receiving $PartialBlock_i$ from ES_m, CS_w will make the full block $Block_i$ on $PartialBlock_i$, by adding the timestamp, unique block version, previous hash block (PBH_i) and current hash block (CBH_i) as shown in Fig. 6, where $CBH_i = h$(Block Header ||Block Payload). $Block_i$ is then mined by a group of cloud servers in a Peer-to-Peer (P2P) CS network that are participated in a consensus algorithm to validate and add that block. We have applied the "Practical Byzantine Fault Tolerance (PBFT)" consensus algorithm [41] for this purpose. Overall, the consensus process is provided in Algorithm 1. The inputs for this algorithm are: a) $Block_i$ = {Block Header, Block Payload, CBH_i}, b) private-public key pairs (pr_{CS_w}, $Pub_{CS_w} = pr_{CS_w} \cdot G$) for all cloud servers CS_w in the P2P CS network, and c) f_{cs}: the number of faulty nodes in the P2P CS network. After successful validation of the block $Block_i$, it is the added in the blockchain. The signature generation and verification are done using the the ECDSA signature. It is worth noticing that the accident data is verified at four stages: 1) verification of "Merkle Tree Root (MTR_i)", 2) verification of the ECDSA signature (Sig_{Block_i}), 3) verification of the "current hash block (CBH_i)", and 4) individual signature (Sig_{Tx_i}) verification of any transaction Tx_i.

E. Dynamic Node Addition Phase

The vehicles in an ITS environment are continuously mobile, and hence, they are subject to changing their locations. Whenever a new vehicle V_i^{new} is added onto the system, the following steps need to be executed:

Step DNA1: For V_i^{new}, the RA picks a unique certificate private and public key pair as ($cpr_{V_i}^{new} \in Z_q^*$, $CPub_{V_i}^{new} = cpr_{V_i}^{new} \cdot G$). The RA then generates a specific identity $ID_{V_i}^{new}$, and calculates its pseudo-identity $RID_{V_i}^{new} = h(ID_{V_i}^{new} ||cpr_{V_i}^{new} ||RTS_{V_i}^{new})$ where $RTS_{V_i}^{new}$ is the registration time of the vehicle V_i^{new} and certificate $Cert_{V_i}^{new} = pr_{RA} + h(RID_{V_i}^{new} ||CPub_{V_i}^{new} ||Pub_{RA}) * cpr_{V_i}^{new} \pmod q$.

Step DNA2: The RA selects a random secret $rs_{V_i}^{new} \in Z_q^*$ to compute the partial-private key $ppr_{V_i}^{new} = h(pr_{RA} ||cpr_{V_i}^{new} ||rs_{V_i}^{new})$ and its respective public key $PK_{V_i}^{new} = ppr_{V_i}^{new} \cdot G$. After that the credentials $\{RID_{V_i}^{new}, Cert_{V_i}^{new}, ppr_{V_i}^{new}\}$ are stored in the on-board unit (OBU) of V_i^{new} that is designed to be tamper-resistant by the RA. The RA also deletes the private key $cpr_{V_i}^{new}$ and makes $PK_{V_i}^{new}$ as public.

Algorithm 1 Consensus Algorithm for a Block ($Block_i$) Verification and Addition in Blockchain

Input: $Block_i$ = {Block Header, Block Payload, CBH_i}; private-public key pairs (pr_{CS_w}, $Pub_{CS_w} = pr_{CS_w} \cdot G$) for all cloud servers CS_w; f_{cs}.

Output: Addition of $Block_i$ in the blockchain after successful validation.

1: Let CS_w select a leader, say CS_L using the existing leader selection algorithm [42].
2: CS_L creates a voting request, say $VReq$.
3: CS_L generates a random nonce r_{CS} and creates a signature Sig_{CS_L} using its own private key pr_{CS_L} on $h(Block_i ||r_{CS} ||VReq ||Cert_{CS_L})$.
4: CS_L sends the request message $\langle Block_i, Sig_{CS_L}, EP_{Pub_{CS_P}}[r_{CS}, VReq], Cert_{CS_L}\rangle$ to other peer cloud servers CS_P in the P2P CS network via public channel.
5: Each peer CS_P after getting request message, computes $(r_{CS}, VReq) = DP_{pr_{CS_P}}[EP_{Pub_{CS_P}}[r_{CS}, VReq]]$, verifies $Cert_{CS_L}$ and Sig_{CS_L} on $Block_i$, r_{CS}, $VReq$ and $Cert_{CS_L}$ using Pub_{CS_L}.
6: If both certificate and signature are valid, CS_P further verifies MTR_i, CBH_i, and Sig_{Block_i} on the received $Block_i$.
7: If all these validations are successful, CS_P prepares the voting response, $VRes$ and generates the ECDSA signature Sig_{CS_P} created on $h(r_{CS} ||VRes ||VReq ||Cert_{CS_P})$ using pr_{CS_P}.
8: CS_P sends the response message $\langle EP_{Pub_{CS_L}}[VRes], Sig_{CS_P}, Cert_{CS_P}\rangle$ to CS_L via public channel.
9: Let $ValidVC$ denote the valid vote counter. Set $ValidVC \leftarrow 0$.
10: **for** each received message $\langle EP_{Pub_{CS_L}}[VRes], Sig_{CS_P}, Cert_{CS_P}\rangle$ from CS_L's followers CS_P **do**
11: CS_L validates $Cert_{CS_P}$ and Sig_{CS_P} using the CS_P's Pub_{CS_P}.
12: CS_L computes $VRes = DP_{pr_{CS_L}}[EP_{Pub_{CS_L}}[VRes]]$.
13: **if** $((Sig_{CS_P} = valid)$ and $(VReq = valid)$ and $(VRes = valid))$ **then**
14: Set $ValidVC = ValidVC + 1$.
15: **end if**
16: **end for**
17: **if** $(ValidVC > 2f_{cs} + 1)$ **then**
18: Send commit response (i.e., $Block_i$ is successfully verified) to all followers CS_P.
19: Add $Block_i$ to the blockchain.
20: **end if**

IV. Security Analysis

In this section, we show that the proposed scheme (BCAS-VADN) is resilient against following potential attacks.

1) Replay Attack: During the authentication phase between a vehicle (V_i) and its CH discussed in Section III-C.1, the messages $Msg_{VCH1} = \langle RID_{V_i}, Cert_{V_i}, X_{V_i}, Sig_{r_{V_i}}, TS_V\rangle$ and

$Msg_{VCH2} = \langle RID_{CH}, Cert_{CH}, Y_{CH}, Sig_{CH1}, TS_{CH1} \rangle$ are being communicated over public channel. Similarly, authentication between a cluster head CH and its associated RSU_l discussed in Section III-C.2, the messages $Msg_{CHRSU_1} = \langle RID_{CH}, Cert_{CH}, S_{CH}, Sign_{r_{CH_2}}, TS_{CH_2} \rangle$, $Msg_{CHRSU_2} = \langle RID_{RSU_l}, Cert_{RSU_l}, T_{RSU}, V_{RSU}, Sign_{r_{RSU}}, TS_{RSU} \rangle$, and $Msg_{CHRSU_3} = \langle SKV_{CH,RSU}, TS_{CH_3} \rangle$ are also transmitted over the open channels. Since the timestamps and random nonces are injected in each message, it provides the freshness of messages by means of validating the timeliness of the received timestamp attached in the message. Therefore, BCAS-VADN is resilient against "replay attack".

2) Man-in-the-Middle(MiTM) Attack: Assume that an adversary $\mathcal{A}$ eavesdrops on the authentication request message Msg_{VCH1}, and tries to build another legitimate message Msg^*_{VCH1} on the fly so that the receiver CH can not detect that it is a modified message. To achieve this goal, $\mathcal{A}$ may generate a timestamp and random number. But without knowledge of the pre-loaded secret ppr_{V_i}, $\mathcal{A}$ can not construct the valid message Msg^*_{VCH1}. Also, $\mathcal{A}$ can not produce valid authentication response message Msg_{VCH2} without knowledge of ppr_{CH}. Similarly, $\mathcal{A}$ fails to produce valid messages Msg_{CHRSU_1}, Msg_{CHRSU_2}, and Msg_{CHRSU_3} without knowledge of secrets ppr_{CH}, pr_{RSU_l} and K_{RSU_l,ES_m}. Hence, BCAS-VADN is resilient against MiTM attack.

3) Impersonation Attacks: Suppose an adversary $\mathcal{A}$ tries to act like a legitimate vehicle V_i. $\mathcal{A}$ may then try to produce an authorized message $Msg^*_{VCH1} = \langle RID_{V_i}, Cert_{V_i}, X^*_{V_i}, Sig_{r_{V_i}}, TS^*_V \rangle$. To establish this goal, $\mathcal{A}$ may pick timestamp TS^*_V and random number $r^*_{V_i} \in Z^*_q$, and compute $X^*_{V_i} = h(RID_{V_i} || ppr_{V_i} || r^*_{V_i} || TS^*_V) \cdot G$. Without knowledge of the secret ppr_{V_i} of V_i, it is a "computationally impossible task" to construct valid $X^*_{V_i}$ and also to generate valid signature $Sig_{r_{V_i}}$. The similar situation occurs when $\mathcal{A}$ attempts to impersonate RSU and ES. Thus, BCAS-VADN is secure against vehicle, RSU and ES "impersonation attacks".

4) Privileged-Insider Attack: In the enrollment phase discussed in Section III-B, no entity sends any registration information to the RA. Instead, the RA only performs the registration process prior to deployment of all the entities. After successfully registration, all the secret credentials (for example, certificate keys, private keys and certificates) are loaded in their respective memory, and then the RA deletes the secret information from its own database corresponding to V_i, RSU_l, ES_m and CS_w. Therefore, an adversary, who is a privileged-insider user, can not obtain any pre-loaded secret information of the deployed entities. Hence, BCAS-VADN is resilient against "privileged-insider attack".

5) Physical Vehicle Capture Attack: Due to the misbehaviour of any vehicle (V_i) user (driver) or any antagonistic environment, a vehicle Vi may be physically seized by an adversary $\mathcal{A}$. $\mathcal{A}$ will be then able to extract all loaded information $\{RID_{V_i}, Cert_{V_i}, ppr_{V_i}\}$ from V_i's OBU_i memory by utilizing the "power analysis attacks" [17]. Since all the loaded credentials in OBU_i memory are unique and distinct from the loaded information in other non-compromised OBU_j, compromise of these information can not help to establish the session keys among other non-compromised vehicles and their respective CHs and RSUs. Thus, other non-compromised vehicles can still communicate with CHs and RSUs with 100% secrecy. This phenomenon is known as "unconditionally secure against vehicle capture attack". Therefore, BCAS-VADN is secure against "physical vehicle capture attack".

6) Ephemeral Secret Leakage (ESL) Attack: In V2CH authentication phase, V_i and its CH establish a common session key $SK_{V_i,CH} = h(DHK_{V_i,CH} || Sig_{r_{V_i}} || Cert_{V_i} || Cert_{CH} || TS_{CH1})$ $(= SK_{CH,V_i})$ for secure communication. Similarly, in CH2RSU authentication phase, CH and RSU_l establish a common session key $SK_{CH,RSU} = h(DHK_{CH,RSU} || \gamma' || Sign_{r_{CH_2}} || Sign_{r_{RSU}} || TS_{CH_3})$ $(= SK_{RSU,CH})$ for secret communication. The session keys are based on both the "session-specific (ephemeral)" secrets (i.e., random secrets) and the "long-term secrets" (i.e., encryption or signature private keys). We have applied the CK-adversary model [16] in the discussed threat model in Section I-A.2. Thus, based on the CK-adversary model, an adversary $\mathcal{A}$ can only obtain the session key between V_i and CH if he/she can obtain both short-term and long-term secrets. This is also true for session key established between CH and its respective RSU_l. Since the session keys utilize the random secrets and timestamps, for every session we will always have different session key. Thus, if a particular session key is revealed for a session, it will not effect in other sessions. Thus, BCAS-VADN provides both perfect "forward and backward secrecy". BCAS-VADN is then resilient against ESL attack.

7) Block Verification in Blockchain: Assume that a verifier $\mathcal{V}$ wants to verify a block $Block_i$ in the blockchain. To verify $Block_i$, $\mathcal{V}$ needs to calculate the "Merkle tree root MTR^*_i" on all the transactions present in that block and then to compute current hash block CBH^*_i on {Block Header, Block Payload}. If $MTR^*_i = MTR_i$ and $CBH^*_i = CBH_i$, $\mathcal{V}$ further verifies Sig_{Block_i} present in the block payload using "ECDSA signature verification algorithm". Only if all the results are positive, $\mathcal{V}$ considers the added block as a legitimate block. Moreover, to modify, delete, or update a block in the blockchain, an adversary requires to change all of the components in the block. Since the signature requires a private key of the signer, for the adversary it is quite impossible to tamper a block $Block_i$.

V. Formal Security Verification: Simulation Study

The "Automated Validation of Internet Security Protocols and Applications(AVISPA)" [43] is a model-checking tool that performs the simulation of network security protocols using a language, called the "High Level Protocol Specification Language (HLPSL)". It has gained huge popularity among the researchers in recent years as it can be easily implemented in many authentication protocols [33], [37], [44]–[48]. HLPSL is based on temporal logic. The simulated protocols can be validated to be either "safe" or "unsafe" or "inconclusive" against "replay attack" and "man-in-the-middle attack". In case, the protocol is deemed to be "unsafe", a trace of the possible attack is presented as a justification, which helps to improvise the protocol. The verification takes place on one of the four available backends provided by AVISPA

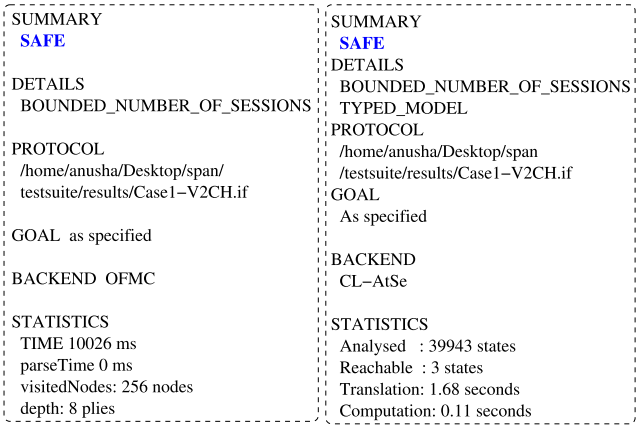


Fig. 7. Simulation results under OFMC and CL-AtSe backends in Case 1.

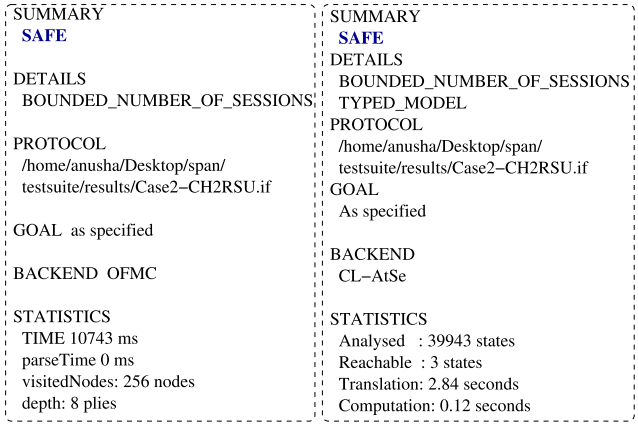


Fig. 8. Simulation results under OFMC and CL-AtSe backends in Case 2.

tool: a) "On The Fly Model Checker (OFMC)", b) "Constraint Logic based Attack Searcher(CL-AtSe)", c) "SAT-based Model Checker(SATMC)", and d) "Tree Automata based on Automatic Approximations for the Analysis of Security Protocols (TA4SP)". HLPSL allows to program the protocol by means of basic and composite roles. Each role comprises various transitions among multiple states. The complete documentation of the working of various backends and the usage of HLPSL for simulation of a security protocol can be found in [43].

We implemented the proposed BCAS-VADN into the following two cases:

Case 1. In this case, we implemented the basic roles for a vehicle (V_i), its associated CH and the RA during the enrollment and authentication phases.

Case 2. Under this case, we implemented the basic roles for a cluster head (CH), its respective RSU_l and the RA during the enrollment and authentication phases.

In both the cases, we implemented the mandatory roles for the "session" and "goal & environment", which are the composite roles of the basic roles. We then simulated both the cases using the "SPAN, the Security Protocol ANimator for AVISPA" [49]. Since AVISPA is based on the DY threat model [15], it assures the replay and man-in-the-middle attacks protection against a tested security protocol. In AVISPA, an intruder (i) always takes an active participation role.

In our HLSPL implementation, under both the cases we performed three verifications related to the testing of the proposed BCAS-VADN, which are: a) "executability checking on non-trivial HLPSL specifications", b) "replay attack checking", and c) "Dolev-Yao (DY) model checking" [15]. An "executability check for non-trivial HLPSL specifications" is needed because due to some modeling mistakes, a protocol model may not complete its execution. This means that the back-ends may not find any attack in a case when the protocol model cannot reach to a state where an attack can occur. As a result, an "executability check" is a very important criteria for formal security verification under the AVISPA tool [50]. For "replay attack checking", the backends verify if "the honest agents can execute a protocol by performing a search of a passive intruder, and then provide the intruder the knowledge of some normal sessions between honest agents" [43]. For the DY model checking, the back-ends check if there is any man-in-the-middle attack that can be performed by the intruder or not. The simulation results demonstrated in both Fig. 7 and 8 show that BCAS-VADN is secure against replay and man-in-the-middle attacks.

TABLE III
COMPARISON OF COMMUNICATION COSTS

Protocol	No. of messages	Total cost (in bits)
BCAS-VADN (Case 1)	2	1856
BCAS-VADN (Case 2)	3	2400
Liu *et al.* [18]	6	11136
Liu *et al.* [19]	3	2208
Li *et al.* [22]	5	5536
Tan and Chung [23] (V2RSU authentication)	$2n+1$	$992+1344n$
Tan and Chung [23] (V2V group key management)	$3m+2$	$3584m$

Note: n: number of neighboring vehicles in a vehicle group in Tan and Chung's scheme; m: number of vehicles willing to be in group in Tan and Chung's scheme [23]

VI. COMPARATIVE ANALYSIS

Under this section, we perform a detailed comparative study on "communication costs", "computational costs" and "security and functionality features" among our proposed BCAS-VADN and other related state-of-art authentication schemes, such as the schemes suggested by Liu *et al.* [18], Liu *et al.* [19], Li *et al.* [22] and Tan and Chung [23]. We consider the authentication-related phase for calculating the communication and computational costs for various schemes. In our BCAS-VADN, we consider two cases: a) *Case 1* for authentication phase between a vehicle and its respective cluster head, and b) *Case 2* for authentication phase between a cluster head and its associated RSU.

1) Communication Costs Comparison: We consider all the communicated messages for authentication in both Case 1 and Case 2. A random nonce, an identity, timestamp, hash

TABLE IV
COMPARATIVE COMPUTATIONAL COSTS ANALYSIS

Scheme	Total cost	Estimated time (in milliseconds)
BCAS-VADN (Case 1)	$10T_h + 12T_{ecm} + 4T_{eca}$	226 ms
BCAS-VADN (Case 2)	$15T_h + 12T_{ecm} + 4T_{eca}$	227.60 ms
Liu *et al.* [18]	$11T_{ecm} + 4T_{eca} + 10T_h + T_{bp}$	251.01 ms
Liu *et al.* [19]	$8T_{bp} + 10T_h + 4T_{kdf} + 5T_{exp} + 2T_{ecc-enc}/T_{ecc-dec} + 6T_{ecm} + 3T_{eca}$	613.26 ms
Li *et al.* [22]	$14T_h + 20T_{ecm} + 8T_{eca}$	381.68 ms
Tan and Chung [23] (V2RSU authentication)	$n(17T_{ecm} + 14T_h + 4T_{bp} + 2T_{eca} + 2T_{exp})$	$510.82n$ ms
Tan and Chung [23] (V2RSU batch authentication)	$(14n+1)T_{ecm} + (13n+1)T_h + 2nT_{eca} + (2n+2)T_{bp} + 3T_{exp}$	$(336.58n + 159.24)$ ms
Tan and Chung [23] (V2V group key management)	$(4m+2)T_h + (5m+1)T_{ecm}$	$(86.78m + 17.74)$ ms

Note: n: number of neighboring vehicles in a vehicle group in Tan and Chung's scheme; m: number of vehicles willing to be in group in Tan and Chung's scheme [23]

output of a one-way cryptographic hash function (using SHA-256 hashing algorithm), and an elliptic curve point $P = (P_x, P_y)$ are assumed to be 160, 160, 32, 256, and $(160 + 160) = 320$ bits, respectively. Moreover, for public key encryption/decryption using ECC, an encrypted message produces 640 bits and a decrypted message becomes 320 bits.

In Case 1 for BCAS-VADN, the communication costs for two messages $Msg_{VCH1} = \langle RID_{V_i}, Cert_{V_i}, X_{V_i}, Sig_{r_{V_i}}, TS_V \rangle$ and $Msg_{VCH2} = \langle RID_{CH}, Cert_{CH}, Y_{CH}, Sig_{CH1}, TS_{CH1} \rangle$ demand equally $(256+160+320+160+32) = 928$ bits, which altogether need 1856 bits. Similarly, in Case 2, three messages $Msg_{CHRSU_1} = \langle RID_{CH}, Cert_{CH}, S_{CH}, Sign_{r_{CH_2}}, TS_{CH_2} \rangle$, $Msg_{CHRSU_2} = \langle RID_{RSU_l}, Cert_{RSU_l}, T_{RSU}, V_{RSU}, Sign_{r_{RSU}}, TS_{RSU} \rangle$, and $Msg_{CHRSU_3} = \langle SKV_{CH,RSU}, TS_{CH_3} \rangle$ demand $(256+160+320+160+32) = 928$ bits, $(256 + 160 + 320 + 256 + 160 + 32) = 1184$ bits, and $(256+32) = 288$ bits respectively, which altogether need 2400 bits. The comparative analysis on communication costs among BCAS-VADN (Case 1 and Case 2) and other schemes shown in Table III clearly demonstrates that the proposed BCAS-VADN needs less communication costs as compared to other schemes.

2) Computation Costs Comparison: We use the existing experimental results reported in [51], [52] for approximated time needed for various cryptographic operations. We denote T_{ecm}, T_{eca}, T_{mtp}, T_{bp}, T_{exp}, T_h, $T_{ecc-enc}/T_{ecc-dec}$ and T_{kdf} as the time required for "ECC point multiplication", "ECC point addition", "map-to-point", "bilinear pairing", "modular exponentiation", "one-way cryptographic hash function", "ECC encryption/decryption", and 'key derivation function (KDF)", respectively. We then have $T_{ecm} \approx 17.10$ ms, $T_{eca} \approx 4.40$ ms, $T_{mtp} \approx 44.06$ ms, $T_{bp} \approx 42.11$ ms, $T_{exp} \approx 19.20$ ms, $T_h \approx 0.32$ ms and $T_{ecc-enc} = 2T_{ecm} + T_{eca} \approx 38.60$ ms, and $T_{ecc-dec} = T_{ecm} + T_{eca} \approx 21.50$ ms. It is assumed that $T_{kdf} \approx T_h$. Furthermore, we discard the time needed to compute modular multiplication over a finite field as it is negligible.

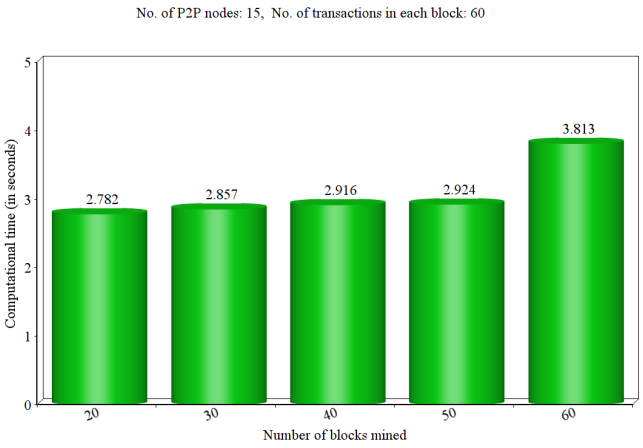


Fig. 9. Blockchain simulation results for Scenario 1.

For our BCAS-VADN, in Case 1, a vehicle (V_i) and its associated CH require the computational costs of $5\ T_h + 6\ T_{ecm} + 2\ T_{eca}$ and $5\ T_h + 6\ T_{ecm} + 2\ T_{eca}$, respectively. The total computational cost for Case 1 is then $10\ T_h + 12\ T_{ecm} + 4\ T_{eca} \approx 226$ ms. Similarly, for Case 2 in BCAS-VADN, CH and its associated RSU_l require the computational costs of $7\ T_h + 6\ T_{ecm} + 2\ T_{eca}$ and $8\ T_h + 6\ T_{ecm} + 2\ T_{eca}$, respectively. Therefore, the total computational cost of Case 2 requires $15\ T_h + 12\ T_{ecm} + 4\ T_{eca} \approx 227.6$ ms. The comparative analysis on computation costs among BCAS-VADN (Case 1 and Case 2) and other schemes tabulated in Table IV also demonstrates that BCAS-VADN needs less computation costs as compared to other schemes.

3) Functionality and Security Features Comparison: Finally, the comparative analysis on various "functionality and security features" among BCAS-VADN (Case 1 and Case 2) and other schemes are tabulated in Table V. It is observed that our proposed BCAS-VADN and another existing scheme of Tan and Chung [23] only support blockchain solution. However, BCAS-VADN provides better security and more functionality featured as compared to those for other schemes.

VII. BLOCKCHAIN IMPLEMENTATION

In this section, we perform the blockchain simulation of the proposed scheme (BCAS-VADN) on a platform having the configuration: "Ubuntu 18.04, 64-bit OS with Intel(R) Core(TM) i5-4210U CPU @ 1.70GHz, 4 GB RAM". The source code was written in Node.js language with VS CODE 2019 [53].

To calculate the total size of the block mentioned in Fig. 6, we have taken the block version, previous block hash, timestamp (epoch time), Merkle tree root, owner of the block, the public key of a signer, block payload, and current block hash (using SHA-256 hashing algorithm) of sizes 32, 256, 42, 256, 160, 320, $(480\ n_t + 320)$, and 256 bits, respectively. Thus, the total size of the block becomes $1386 + 480\ n_t$, where n_t is the total number of transactions stored in a block.

TABLE V
COMPARISON OF FUNCTIONALITY & SECURITY FEATURES

Feature	Liu *et al.* [18]	Liu *et al.* [19]	Li *et al.* [22]	Tan and Chung [23]	BCAS-VADN
Insider attack	×	✓	✓	×	✓
Replay attack	✓	✓	✓	✓	✓
Man-in-the-middle attack	✓	✓	✓	✓	✓
Mutual authentication	✓	✓	✓	✓	✓
Key agreement	✓	✓	✓	✓	✓
Vehicle impersonation attack	✓	✓	✓	✓	✓
RSU impersonation attack	✓	✓	✓	✓	✓
Edge server impersonation attack	×	×	×	×	✓
Session key security under the CK-adversary model	×	×	✓	×	✓
Formal security verification using AVISPA tool	×	×	×	×	✓
Dynamic vehicle addition phase	×	×	×	×	✓
Support to blockchain-based solution	×	×	×	✓	✓

Note: ✓: a scheme is secure or assists a feature; ×: a scheme is insecure or does not assist a feature.

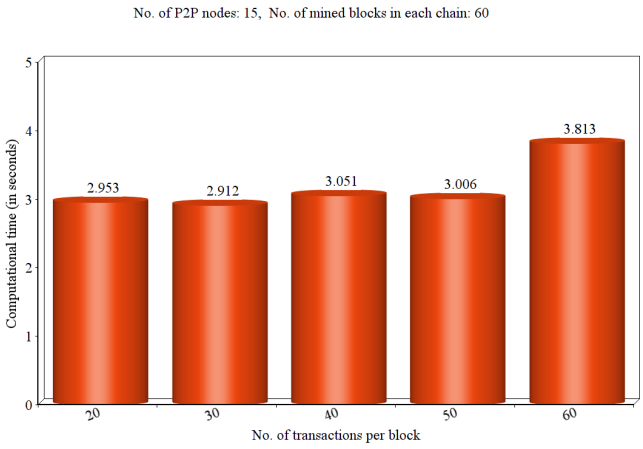


Fig. 10. Blockchain simulation results for Scenario 2.

In the following, we consider the following two types of simulation scenarios:

- **Scenario 1:** We assume that the total number of peer-to-peer (P2P) nodes in the CS network is 15. The simulation results for this scenario are shown in Fig. 9. It is worth noticing that the average computational time (in seconds) per block differs for the varied number of blocks mined into a blockchain, where each block contains a fixed number of transactions as 60.
- **Scenario 2:** In this situation, the total number of peer-to-peer (P2P) nodes in CS network is also considered as 15. Each created blockchain has a fixed number of blocks as 60. The simulation results reported in Fig. 10 shows the average computational time (in seconds) per block also differs based on the number of varied transactions pushed per block.

VIII. CONCLUSION

In this work, we looked into an interesting problem for vehicle accident detection and notification in an ITS environment. To handle this problem, we designed a new blockchain-enabled certificate-based authentication scheme (BCAS-VADN) so that a vehicle can report securely the transactions related to accident detection and notification of its own or neighboring vehicles to its cluster head. The cluster head then securely communicates with its respective RSU and forwards securely the transactions to its associated edge server(s). The edge server analyzes the transactions, and valuable transactions contribute to construct partial blocks, and it sends the partial blocks to its cloud server in the BC. The cloud server completes the partial blocks into full blocks and adds them into the BC after running the proposed consensus algorithm. The secret keys establishment among various participants happen via the proposed authentication mechanisms. The security analysis shows that BCAS-VADN can resist several potential attacks. Moreover, a detailed comparative study among BCAS-VADN and other related state-of-art schemes exhibits superiority of BCAS-VADN over other schemes in terms of less communication and computational overheads, and better security and support to more functionality attributes. The practical demonstration of our proposed BCAS-VADN using the blockchain technology has been also provided to show the computational time for the varied number of blocks mined and the varied number of transactions per block.

ACKNOWLEDGMENT

The authors thank the anonymous reviewers and associate editor for their valuable feedback on the paper, which helped them to improve its quality and presentation.

REFERENCES

[1] (2020). *Violence and Injury Prevention*. World Health Organization (WHO). Accessed: Feb. 2020. [Online]. Available: https://www.who.int/violence_injury_prevention/road_traffic/en/
[2] J. J. Rolison, S. Regev, S. Moutari, and A. Feeney, "What are the factors that contribute to road accidents? An assessment of law enforcement views, ordinary drivers' opinions, and road accident records," *Accident Anal. Prevention*, vol. 115, pp. 11–24, Jun. 2018.
[3] (2020). *Communication for eSafety (COMeSafety)*. Accessed: Mar. 2020. [Online]. Available: https://trimis.ec.europa.eu/project/communications-esafety
[4] (2012). *OnStar by GM*. Accessed: Mar. 2020. [Online]. Available: https://www.onstar.com/us/en/home/
[5] M. Fogue, P. Garrido, F. Martinez, J.-C. Cano, C. Calafate, and P. Manzoni, "Automatic accident detection: Assistance through communication technologies and vehicles," *IEEE Veh. Technol. Mag.*, vol. 7, no. 3, pp. 90–100, Sep. 2012.

[6] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[7] (2019). *Establishing the Edge: A New Infrastructure Model for Service Providers–Cisco [Whitepaper]*. [Online]. Available: https://www.cisco.com/c/en/us/solutions/service-provider/edge-computing/establishing-the-edge.html

[8] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.

[9] S. R. Pokhrel and J. Choi, "Federated learning with blockchain for autonomous vehicles: Analysis and design challenges," *IEEE Trans. Commun.*, early access, Apr. 27, 2020, doi: 10.1109/TCOMM.2020.2990686.

[10] B. McMahan and D. Ramage. (2017). *Federated Learning: Collaborative Machine Learning Without Centralized Training Data*. Accessed: Jun. 2020. [Online]. Available: https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

[11] J. Konečný, H. Brendan McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016, *arXiv:1610.02527*. [Online]. Available: http://arxiv.org/abs/1610.02527

[12] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Blockchain empowered asynchronous federated learning for secure data sharing in Internet of vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4298–4311, Apr. 2020.

[13] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4177–4186, Jun. 2020.

[14] A. Jolfaei and K. Kant, "Data security in multiparty edge computing environments," in *Proc. Government Microcircuit Appl. Crit. Technol. Conf., Artif. Intell. Cyber Secur., Challenges Opportunities Government*, Albuquerque, NM, USA, 2019, pp. 17–22.

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

[16] R. Canetti and H. Krawczyk, "Universally composable notions of key exchange and secure channels," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, Amsterdam, The Netherlands, 2002, pp. 337–351.

[17] T. S. Messerges, E. A. Dabbish, and R. H. Sloan, "Examining smart-card security under the threat of power analysis attacks," *IEEE Trans. Comput.*, vol. 51, no. 5, pp. 541–552, May 2002.

[18] Y. Liu, Y. Wang, and G. Chang, "Efficient privacy-preserving dual authentication and key agreement scheme for secure V2 V communications in an IoV paradigm," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 10, pp. 2740–2749, Oct. 2017.

[19] J. Liu, Q. Li, R. Sun, X. Du, and M. Guizani, "An efficient anonymous authentication scheme for Internet of vehicles," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–6.

[20] L. Wu *et al.*, "An efficient privacy-preserving mutual authentication scheme for secure V2 V communication in vehicular ad hoc network," *IEEE Access*, vol. 7, pp. 55050–55063, 2019.

[21] Q. Jiang, X. Zhang, N. Zhang, Y. Tian, X. Ma, and J. Ma, "Two-factor authentication protocol using physical unclonable function for IoV," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, Changchun, China, Aug. 2019, pp. 195–200.

[22] X. Li, Y. Han, J. Gao, and J. Niu, "Secure hierarchical authentication protocol in VANET," *IET Inf. Secur.*, vol. 14, no. 1, pp. 99–110, Jan. 2020.

[23] H. Tan and I. Chung, "Secure authentication and key management with blockchain in VANETs," *IEEE Access*, vol. 8, pp. 2482–2498, 2020.

[24] M. Barbareschi, P. Bagnasco, and A. Mazzeo, "Authenticating IoT devices with physically unclonable functions models," in *Proc. 10th Int. Conf. P2P, Parallel, Grid, Cloud Internet Comput. (3PGCIC)*, Kraków, Poland, Nov. 2015, pp. 563–567.

[25] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith, "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data," *SIAM J. Comput.*, vol. 38, no. 1, pp. 97–139, Jan. 2008.

[26] C. P. Schnorr, "Efficient identification and signatures for smart cards," in *Proc. Conf. Theory Appl. Cryptol.*, Santa Barbara, CA, USA, 1990, pp. 239–252.

[27] (2018). *Global Status Report on Road Safety*. World Health Organization (WHO). Accessed: Feb. 2020. [Online]. Available: https://www.who.int/violence_injury_prevention/road_safety_status/2018/en/

[28] (2017). *Accidental Deaths and Suicides in India*. National Crime Records Bureau, Ministry of Home Affairs, Govt. of India. Accessed: Mar. 2020. [Online]. Available: http://ncrb.gov.in/StatPublications/ADSI/ADSI2017/ADSI2017.html

[29] C. Lin, D. He, N. Kumar, K.-K.-R. Choo, A. Vinel, and X. Huang, "Security and privacy for the Internet of drones: Challenges and solutions," *IEEE Commun. Mag.*, vol. 56, no. 1, pp. 64–69, Jan. 2018.

[30] M. Wazid, A. K. Das, N. Kumar, M. Conti, and A. V. Vasilakos, "A novel authentication and key agreement scheme for implantable medical devices deployment," *IEEE J. Biomed. Health Informat.*, vol. 22, no. 4, pp. 1299–1309, Jul. 2018.

[31] N. Kumar, S. Misra, J. J. P. C. Rodrigues, and M. S. Obaidat, "Coalition games for spatio-temporal big data in Internet of vehicles environment: A comparative analysis," *IEEE Internet Things J.*, vol. 2, no. 4, pp. 310–320, Aug. 2015.

[32] D. He, N. Kumar, and J.-H. Lee, "Privacy-preserving data aggregation scheme against internal attackers in smart grids," *Wireless Netw.*, vol. 22, no. 2, pp. 491–502, Feb. 2016.

[33] A. Dua, N. Kumar, A. K. Das, and W. Susilo, "Secure message communication protocol among vehicles in smart city," *IEEE Trans. Veh. Technol.*, vol. 67, no. 5, pp. 4359–4373, May 2018.

[34] C.-C. Chang and H.-D. Le, "A provably secure, efficient, and flexible authentication scheme for ad hoc wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 1, pp. 357–366, Jan. 2016.

[35] M. Wazid *et al.*, "Design of lightweight authentication and key agreement protocol for vehicular ad hoc networks," *IEEE Access*, vol. 5, pp. 14966–14980, 2017.

[36] A. K. Das, M. Wazid, N. Kumar, A. V. Vasilakos, and J. J. P. C. Rodrigues, "Biometrics-based privacy-preserving user authentication scheme for cloud-based industrial Internet of Things deployment," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4900–4913, Dec. 2018.

[37] J. Srinivas, A. K. Das, N. Kumar, and J. J. P. C. Rodrigues, "TCALAS: Temporal credential-based anonymous lightweight authentication scheme for Internet of drones environment," *IEEE Trans. Veh. Technol.*, vol. 68, no. 7, pp. 6903–6916, Jul. 2019.

[38] M. S. Kakkasageri and S. S. Manvi, "Multiagent driven dynamic clustering of vehicles in VANETs," *J. Netw. Comput. Appl.*, vol. 35, no. 6, pp. 1771–1780, Nov. 2012.

[39] A. Jolfaei, K. Kant, and H. Shafei, "Secure data streaming to untrusted road side units in intelligent transportation system," in *Proc. 18th IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun./13th IEEE Int. Conf. Big Data Sci. Eng. (TrustCom/BigDataSE)*, Rotorua, New Zealand, Aug. 2019, pp. 793–798.

[40] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ECDSA)," *Int. J. Inf. Secur.*, vol. 1, no. 1, pp. 36–63, Aug. 2001.

[41] M. Castro and B. Liskov, "Practical byzantine fault tolerance and proactive recovery," *ACM Trans. Comput. Syst. (TOCS)*, vol. 20, no. 4, pp. 398–461, Nov. 2002.

[42] H. Zhang, J. Wang, and Y. Ding, "Blockchain-based decentralized and secure keyless signature scheme for smart grid," *Energy*, vol. 180, pp. 955–967, Aug. 2019.

[43] (2006). *Automated Validation of Internet Security Protocols and Applications*. Accessed: Mar. 2020. [Online]. Available: http://www.avispa-project.org/

[44] A. K. Das, S. Kumari, V. Odelu, X. Li, F. Wu, and X. Huang, "Provably secure user authentication and key agreement scheme for wireless sensor networks," *Secur. Commun. Netw.*, vol. 9, no. 16, pp. 3670–3687, Nov. 2016.

[45] M. Wazid, A. K. Das, S. Kumari, X. Li, and F. Wu, "Design of an efficient and provably secure anonymity preserving three-factor user authentication and key agreement scheme for TMIS," *Secur. Commun. Netw.*, vol. 9, no. 13, pp. 1983–2001, Sep. 2016.

[46] S. Challa *et al.*, "An efficient ECC-based provably secure three-factor user authentication and key agreement protocol for wireless healthcare sensor networks," *Comput. Electr. Eng.*, vol. 69, pp. 534–554, Jul. 2018.

[47] J. Srinivas, A. K. Das, N. Kumar, and J. Rodrigues, "Cloud centric authentication for wearable healthcare monitoring system," *IEEE Trans. Dependable Secure Comput.*, early access, Apr. 19, 2018, doi: 10.1109/TDSC.2018.2828306.

[48] M. Wazid, A. K. Das, V. Odelu, N. Kumar, and W. Susilo, "Secure remote user authenticated key establishment protocol for smart home environment," *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 2, pp. 391–406, Mar. 2020.

[49] (2020). *SPAN, the Security Protocol ANimator for AVISPA*. Accessed: Mar. 2020. [Online]. Available: http://www.avispa-project.org/

[50] D. von Oheimb, "The high-level protocol specification language HLPSL developed in the EU project AVISPA," in *Proc. 3rd Workshop Appl. Semantic (APPSEM)*, Frauenchiemsee, Germany, 2005, pp. 1–17.

[51] C.-C. Lee, C.-T. Chen, P.-H. Wu, and T.-Y. Chen, "Three-factor control protocol based on elliptic curve cryptosystem for universal serial bus mass storage devices," *IET Comput. Digit. Techn.*, vol. 7, no. 1, pp. 48–56, Jan. 2013.
[52] D. He, S. Zeadally, B. Xu, and X. Huang, "An efficient identity-based conditional privacy-preserving authentication scheme for vehicular ad hoc networks," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 12, pp. 2681–2691, Dec. 2015.
[53] K. Khullar. (2019). *Implementing PBFT in Blockchain: Implementation of PBFT in Node.js*. Accessed: Mar. 2020. [Online]. Available: https://medium.com/coinmonks/implementing-pbft-in-blockchain-12368c6c9548

Anusha Vangala received the M.Tech. degree in computer science and engineering from Jawaharlal Nehru Technological University, Kakinada, India. She is currently pursuing the Ph.D. degree in computer science and engineering with the Center for Security, Theory and Algorithmic Research, IIIT Hyderabad, India. Prior to joining at IIIT Hyderabad for Ph.D. program, she had nearly five years of teaching experience as an Assistant Professor of Computer Science and Engineering at various renowned institutes across India. Her research interests include cloud computing, the Internet of Things (IoT), and blockchain technology.

Basudeb Bera received the M.Sc. degree in mathematics and computing from IIT (ISM), Dhanbad, India, in 2014, and the M.Tech. degree in computer science and data processing from IIT Kharagpur, India, in 2017. He is currently pursuing the Ph.D. degree in computer science and engineering with the Center for Security, Theory and Algorithmic Research, IIIT Hyderabad, India. His research interests are cryptography, network security, and blockchain technology. He has published five papers in international journals and conferences in his research areas.

Sourav Saha (Student Member, IEEE) received the B.Tech. degree in computer science and engineering from the Central Institute of Technology, Kokrajhar, India, and the M.S. degree in computer science and engineering from the Indian Institute of Information Technology, Sri City, Chittoor, India. He is currently pursuing the Ph.D. degree in computer science and engineering with the Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad, India. His research interests include network security and blockchain technology. He has published five papers in international journals and conferences in his research areas.

Ashok Kumar Das (Senior Member, IEEE) received the M.Tech. degree in computer science and data processing, the M.Sc. degree in mathematics, and the Ph.D. degree in computer science and engineering from IIT Kharagpur, India. He is currently an Associate Professor with the Center for Security, Theory and Algorithmic Research, IIIT, Hyderabad, India. His current research interests include cryptography and network security including security in smart grid, the Internet of Things (IoT), the Internet of Drones (IoD), the Internet of Vehicles (IoV), cyber-physical systems (CPS) and cloud computing, and blockchain. He has authored over 225 papers in international journals and conferences in the above areas, including over 195 reputed journal articles. He was a recipient of the Institute Silver Medal from IIT Kharagpur. He is on the editorial board of *KSII Transactions on Internet and Information Systems*, the *International Journal of Internet Technology and Secured Transactions* (Inderscience), and *IET Communications*, is a Guest Editor of *Computers and Electrical Engineering* (Elsevier) for the special issue on Big data and IoT in e-healthcare and of *ICT Express* (Elsevier) for the special issue on Blockchain Technologies and Applications for 5G Enabled IoT, and has served as a program committee member in many international conferences. He also severed as one of the Technical Program Committee Chairs of the International Congress on Blockchain and Applications (BLOCKCHAIN'19), Avila, Spain, June 2019.

Neeraj Kumar (Senior Member, IEEE) received the Ph.D. degree in CSE from Shri Mata Vaishno Devi University, Katra (J & K), India. He was a Postdoctoral Research Fellow with Coventry University, Coventry, U.K. He is currently working as a Full Professor with the Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology (Deemed to be University), Patiala (Pb.), India. He has published more than 400 technical research papers in leading journals and conferences from IEEE, Elsevier, Springer, and John Wiley. He is on the editorial board of *ACM Computing Survey*, the IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING, the *IEEE Network Magazine*, the *IEEE Communication Magazine*, the *Journal of Networks and Computer Applications* (Elsevier), *Computer Communications* (Elsevier), the *International Journal of Communication Systems* (Wiley), and *Security and Privacy* (Wiley).

Youngho Park (Member, IEEE) received the B.S., M.S., and Ph.D. degrees from electronic engineering from Kyungpook National University, Daegu, South Korea, in 1989,1991, and 1995, respectively. He is currently a Professor with the School of Electronics Engineering, Kyungpook National University. From 1996 to 2008, he was a Professor with the School of Electronics and Electrical Engineering, Sangju National University, South Korea. From 2003 to 2004, he was a Visiting Scholar with the School of Electrical Engineering and Computer Science, Oregon State University, USA. His research interests include computer networks, multimedia, and information security.