A vehicle-to-infrastructure communication privacy protocolused Blockchain

Samira Harrabi

samira.harrabi@gmail.com

University of Kairouan

Ines Ben Jaafar

University of Kairouan

Oumaima Omrani

University of Kairouan

Research Article

Keywords: Internet of vehicles, Security , Blockchain, Certificate Authority, Cryptography

Posted Date: March 21st, 2024

DOI: https://doi.org/10.21203/rs.3.rs-4078147/v1

Additional Declarations: No competing interests reported.

A vehicle-to-infrastructure communication privacy protocolused Blockchain

Samira Harrabi[1*],ISIGK University of Kairouan, Tunisia
Ines Ben Jaafar[2] ,ENCT University of Manouba, Tunisia
Oumaima Omrani[3]ENCT University of Manouba, Tunisia

Emails: Corresponding Author:samira.harrabi@gmail.com,

Abstract

Since several decade, the Internet of Things IoT has attracted enormous interest in the research community and industry. However, IoT technologies has completely transformed vehicular ad hoc networks (VANETs) into the "Internet of Vehicles" IoV. In IoV networks, we need to integrate many different technologies, services and standards. However, the heterogeneity and large number of vehicles will increase the need of data security. The IoV security issues are critical because of the vulnerabilities that exist during the transmission of information that expose the IoV to attacks. Each attack has a security procedure. Many protocols and mechanisms exist to combat or avoid this communication security problem. One of these protocols is VIPER (a Vehicle-to-Infrastructure communication Privacy Enforcement pRotocol). In our work, we try to improve this protocol by using Blockchain technology and certification authority.

Keywords: Internet of vehicles, Security , Blockchain, Certificate Authority, Cryptography

1.Introduction

Recently, The Internet of Vehicles (IoV) has been proposed as a significant improvement in the area of vehicular communications. It might be considered as a global reach of a vehicular network. The Internet of Vehicles has become a specific application of the Internet of Things. In the IoV, the possibility of connected and autonomous vehicles will emerge. In effect, in this paradigm, vehicles will be able to communicate with each other and with their environments: pedestrians, traffic, devices, signs, etc. As a result, effective applications for road safety and overall traffic efficiency will be implemented and deployed, thus reducing traffic accidents.There are different types of communication in IoV : vehicle-to-vehicle V2V, vehicle-infrastructure V2I, in general, vehicle-to-everything V2X which is also depicted in Fig.1[12, 13]. The IoV offers multiple benefits, like an integrated warning system that alerts drivers to accidents so that they can make a decision quickly based on the road information provided. More sensitive information can eventually be shared between vehicles and improve the security and accuracy of self-driving vehicles. However, in the absence of the effective security and privacy measures, an adversary can easily collect the data transmitted through networks which usually include the private data of vehicles' users. In addition to

privacy issues, the integrity or authenticity of the data is an important security issue in IoV [12, 19].

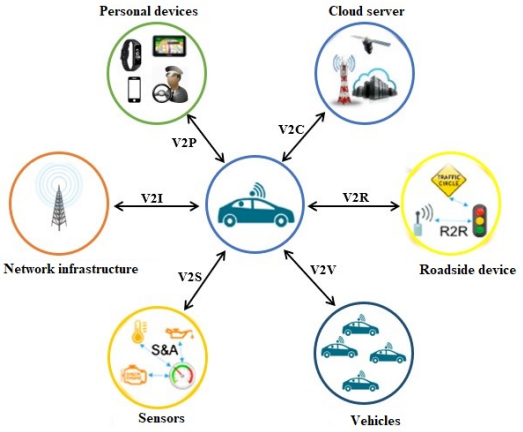

Figure 1: Types of communication in IoV

.

Recently, many protocols for IoV or for VANET have been designed to protect the security of vehicles. In 2008, Ying et al. proposed an vehicle to infrastructure privacy enforcement protocol called VIPER [4]. However , we found that this protocol has problem at registration phase of vehicles and computation during decryption is time consuming. To address these shortcomings, we propose an amelioration of this protocol to solve the above mentioned problems. According to our performance and the security analysis, the proposed protocol is more efficient compared with VIPER protocol.

The remainder of the paper is organized as follows. In Section 2, the related works are described. Section 3 introduces the background of blockchain technology. Next, in section 4 we present our proposed protocol. In Section 5, we analyze the security and overhead of the proposed authentication scheme. Finally, the conclusions are described in Section 6.

2.RELATED WORK

Recently, Many researches have been performed to improve the security in the vehicular networks. In general, confidentiality, integrity, and authenticity are the important aspects for the security of vehicular networks. In 2019, Zhang and al.[23] have proposed an algorithm for detecting the sinkholes in wireless sensor networks based on the random routes selected by the minimum-hop RMHSD. The RMHSD consists of a several step approach: Clustering the network, establishing the hop database, establishing the minimum-hop paths, and computing the hop difference. The new measure called the frequency of each node by establishing M routes with optimum hops is presented on the basis of dynamic programming.However, this approach has difficulty in detecting Sinkhole attacks when there are multiple attacks. Yao and al.[22] proposed a received signal strength indicator (RSSI) based on Sybil detection method, called Voiceprint, which is based on RSSI time series as vehicular speech and performs the comparison between all received time series, as opposed to other RSSI methods that were based on the absolute or relative distance according to RSSI values. In order to improve the observation time and reduce the rate of false positives, Voiceprint has been improved by allowing it to perform detection on the service channel (SCH) and also efforts have been made to detect the malicious nodes performing power control using the change point detection method. However, this proposed solution for power control remains a complex problem when it is adopted with an RSSI-based detection scheme. Also, in 2017, Feng et Tang [7] proposed a method called Sybil attack detection with the obscured neighbor relationship of roadside units (DMON) in order to detect the malicious nodes in the Sybil attack. Therefore, this paper focuses on the following four objectives of the proposed framework: first, to detect malicious Sybil nodes using a ring signature based identification scheme to generate a signed certificate as a temporal identity and using the neighborhood relation to detect malicious nodes, the second objective is to maintain the privacy information of the vehicle by obfuscating the neighborhood relation of RSUs, the third objective is the online and independent realization. Last objective is to achieve high efficiency and low overhead. However, DMON needs to synchronize the information of all RSUs during signature verification. In 2014, Chang and Lee [14] implemented a lightweight, decentralized authentication scheme called Trust-Extended Authentication Mechanism (TEAM) for use in vehicle-to-vehicle communication in VANETs. TEAM consists of public authentication and trust-based authentication. TEAM satisfies security requirements including anonymity, location privacy, mutual authentication, resistance to forgery attacks, resistance to tampering attacks, resistance to replay attacks, no clock synchronization issues, absence

of a verification table, and rapid error detection and resistance to man-in-the-middle attacks. In [20], the authors Vijayakumar et al. proposed a privacy preserving and anonymous authentication scheme using anonymous certicates. In addition, it provides a batch verification for RSU-based vehicle group authentication. Although, it is still not enough efficient due to the communication overhead and the message loss ratio are not considered, and it is vulnerable to the DoS attack. Kong et al. [9] applied a new efficient location privacy preserving data sharing scheme using homomorphic encryption and proxy re-encryption technique. But, it makes loss in the energy consumption calculation due to the re-encryption of sensing data aggregation.

3.Preliminary Knowledge

3.1.Blockchain

Blockchain is one of the more promising technologies of the future, capable of overcoming the aforementioned obstacles [2, 17]. It effectively replaces the current transaction system. The concept of blockchain was started by a researcher with the pseudonym "Satoshi Namakato", who implemented this technology for the implementation of the crypto-currency Bitcoin in 2008. This technology is decentralized and uses a distributed public ledger where the blocks are encrypted and chained in chronological order. There are various features of blockchain, such as proof-of-work and the public ledger concept, that can be used independently and applied in different sectors. [1, 2].

At present, researchers have used blockchain to solve privacy problems on the Internet of Vehicles. In [15], Pokhrel and Choi implemented the blockchain and federated learning in autodriving for protecting users' privacy and used a blockchain incentive mechanism to award the worker nodes with best performance in federated learning in order to encourage them to more actively participate in federated learning. Another example is Liu et al. proposed a blockchain and smart contract-based energy trading model for electric vehicles [10]. Also in [16], Rawat applied blockchain to V2X communication in order to protect data privacy. Das et al. applied the blockchain technology to vehicle theft prevention to provide both vehicle security and owner privacy using smart contracts[6]. Therefore, the registrations of information in the blockchain have the characteristics of a lifetime responsibility system. Once completed, it becomes very difficult to change and delete. With the deterrence power of this lifetime accountability system, the business cooperation, the social behavior and the credibility will be highly improved. And it will be of great help in increasing the construction of our future system of credit and even the progress of human civilization. Finally, the Blockchain technology ensures that all parts of the collaboration view the same information system, which provides a good basis to build a wider range of social cooperation in the future.

In this paper, all roadside units are used as peer nodes to build a blockchain network, which is responsible for collective maintenance of blockchain data. The blockchain network mainly stores the vehicle identity information that contains the hash value of the pseudonym, public key of the vehicle and the mapping relationship between the pseudonym and the real identity of the vehicle.

3.2.Consensus Algorithm

The consensus algorithm is the most important part of the Blockchain technology. It allows distributed nodes to maintain the same blockchain. The principal objective of the consensus algorithm is to make sure that each node checks the generation of blocks in a distributed manner. However, there are three typical consensus algorithms: proof of work (PoW), proof of stake (PoS) and Practical Byzantine Fault Tolerance (PBFT) [1]. The PBFT consensus algorithm uses a two-thirds majority voting procedure. This algorithm is mostly used in private blockchains, and IBM hyperledger fabric is a typical example. The procedure of PBFT consensus algorithm is composed of five parts: REQUEST, PRE-PREPARE,PREPARE, COMMIT and REPLY. Figure 2 shows the global procedure of the Practical Byzantine Fault Tolerance algorithm. In the PBFT system, every node can access the public key of other nodes. Therefore, each node can determine who sent a transaction. There are two types of nodes, a primary node (N0) and replica nodes (N1, N2, N3). The primary node N0 is selected during the leader selection process or when it receives a message from client C for the first time. The replica node is the other nodes (N1,N2,N3) in the cluster except the primary node. During the request process, C broadcasts a transaction to all other nodes. The primary node N0 generates a block with the received transactions and broadcasts the block in the next process "prepara-

tion". In the preparation and validation process, each replica node will confirm whether it received the same block or not. Then they check that the transactions and values in the block are correct. Finally, all nodes send the result of the verification to the client in a response process. [3]

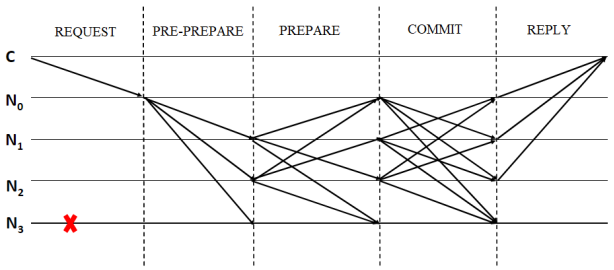

Figure 2: Procedure of the PBFT algorithm

4. PROPOSED APPROACH

This section details our protocol; the intuition behind this protocol is to improve the VIPER protocol [4] to reduce the risk of attacks when registering an OBU in RSU. In this mechanism, vehicles do not send their messages directly to the RSU, but rather vehicles act as mixtures [4, 5]; furthermore, messages are encrypted via a CRT-RSA cryptography algorithm. Mixing is restricted to nodes belonging to the same group [4], where a group is defined as the set of registered vehicles within a roadside unit (the LVC list, for more details on vehicle registration). We show that by combining techniques (CRT-RSA asymmetric cryptography algorithm, blockchain, and the certificate authority), the protocol is resilient against some traffic analysis attacks and other types of attacks while preserving its scalability. The notions used in this paper are listed in TABLE 1.

Table 1: Notations

Notions	Definition
V	set of vehicle
R	set of RSU
S	set of cloud server
PK_Vi	set of the public key of vehicle
SK_Vi	set of the private key of vehicle
PK_R	set of the public key of RSU
SK_R	set of the private key of RSU
SV	set the random value
Verify_Msg_Sign	Verify the signed message with the real message with PK
sign_Msg	sign with the private key
encrypt()	encrypt with the public key
chooseRelay()	Choose a relay vehicle according to the GR group
choose()	select message from buffer

4.1 Routing

When a vehicle Vi needs to send a message m to the RSU, it flips a biased coin, where the faces represent "forward" (with the associated probability pf > 1/2) and "send" (with the associated probability 1-pf). If the result of the coin toss is "forward", the VIPER randomly selects another vehicle V_j (the latter will be called a relay) from the same group to which it belongs (list of relay vehicles, Section 4.3), and this vehicle becomes the recipient of message m. The vehicle V_i can also choose itself as a relay. On the other hand, if the result of the draw is "send", the RSU R_k is the recipient of the message m [4]. The figure below shows an example of routing in this protocol [4].

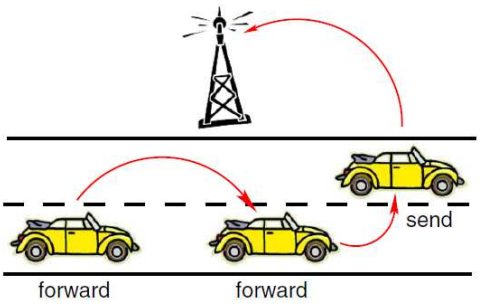

Figure 3: example of routing.

When a relay node (vehicle V_j) receives a message m (this message is in the form as described in Section 4.3- figure 4), it performs the same operations, i.e., it can choose as its next hop another randomly chosen vehicle with probability pf, or the RSU with probability

1-pf . [4]. The operations performed by a vehicle upon receiving a message are summarized in Algorithm 1.

Algorithm 1. Operations performed for the generation of message to RSU.

Input: M_R message to RSU, Q queue, R RSU, GR relay vehicle group, PK_RSU public key of RSU, SK_V private key of the vehicle V
Ouput:

encrypt_message_R ← encrypt(M_R, PK_RSU);
Coin ← flip();
If Coin == forward **then**
    Next_hop ← chooserelay (GR);
    MC ← encrypt (concat (encrypt_message_R, PID_V_R), PK_V_R);
Else
    Next_hop ← R;
    MC ← encrypt_message_R;
End
singed_message ← sign_Msg(MC, SK_V)
M ← former_message (MC,singed_message, Next_hop);
Q ← Q ∪ M;

When a message m is received, V (i.e., the relay vehicle) must verify this message (according to algorithm 4). Then, if this verification is true, then V decrypts the encrypted message (The element "message " in figure 4) by its private key SK_V (according to the decryption algorithm of CRT-RSA). The result of this decryption is as follows: encrypted message (encrypted_message_R) + PID_V. According to the next hop which is decided locally as described in Algorithm 2, if the next hop is a relay vehicle VR, then V concatenates "encrypted_message_R" with the PID_VR and encrypts this concatenation, and then signs the result of this encryption by its private key SK_V. In case the next hop is an RSU, V signs the encrypted message "encrypted_message_R". The use of the encryption operation on the concatenation of the encrypted message with the pseudo-identity of the relay vehicle (PID_VR) plays an essential role in the resilience of our approach against the message encryption attack (Section 5.1). The operations performed by a vehicle upon receiving a message are summarized in Algorithm 2.

At the end of each time interval, each vehicle must send a batch of n messages. Note that the message size, batch size and time interval length are fixed. The vehicles act as mixed nodes, i.e., at the end of the time interval, a subset of messages in the queue is selected to be sent [4]. This selection is done according to a specific policy - for example, random selection, First In First Out (FIFO) , Last In First Out (LIFO) , etc. Then, the sent messages are removed from the queue. To avoid a particular type of traffic analysis, i.e. the volume of messages

Algorithm 2 . Operations performed at the end of slot.

Input: M encrypted message, Q queue, q queue size, R RSU, GR relay vehicle group, SK_V private key of the vehicle
Ouput: Handling of the received message

If $|Q| + |M| \geq$ q ou $|Q|$ = q **then**
    * increase the maximum queue size q */
End
Coin ← flip();
If Coin == forward **then**
    Next_hop ← chooserelay (GR);
    encrypt_Msg ← encrypt (concat (M, PID_V_R), PK_V_R);
Else
    Next_hop ← R;
    encrypt_Msg ← M;
End
singed_message ← sign_Msg(encrypt_Msg, SK_V)
Msg ← former_message (encrypt_Msg,singed_message, Next_hop);
Q ← Q ∪ Msg;

on attack, the batch size should be fixed [4]. To do this, if the number of messages in the queue is less than the batch size, VIPER adds dummy messages to the queue [4]. The operations performed by a vehicle at the end of each time interval are summarized in Algorithm 3.

Algorithm 3 . Operations performed

Input: Q queue, batch size n
Ouput: sending of a batch of n messages

batch ← vide ;
If $|Q| < n$ **then**
    For i ← 1 to n - $|Q|$ **do**
        * Add dummy messages at Q */
    End
    batch ← Q;
Else
    Pour i ← 1 to n **faire**
        M ← choose (Q) ;
        * select a message from Q according to a certain policy */
        batch ← batch ∪ M;
    End
End
send (batch); * send n the batch in random order */
Q ← Q ? batch ; * remove sent messages from Q */

4.2 Encryption

We use the CRT-RSA cryptographic algorithm [18], while sending a message from a vehicle Vi to another node (RSU or a relay vehicle VR) , which is encrypted by the public key PK of V or RSU (according to the encryption algorithm of CRT-RSA [18]). When receiving the message from the vehicle in RSU or in VR, and after the verification (according to algorithm 4), RSU or VR can decrypt the message by its private key Sk_R or SK_VR (according to the decryption algorithm of CRT-

RSA [18]) . The node V (i.e. the sender) must encrypt the message that will be sent to the RSU (receiver) by the RSU public key PK_R. And this message will be in this format in the figure 4. Then, if the next receiver of this message is a relay vehicle VR (according to the third step of algorithm 2), V concatenates the encrypted message with the pseudo-identity of the relay vehicle and encrypts this concatenation with the public key of VR (according to the encryption algorithm of CRT-RSA [18]). Then, it signs this encrypted message with the private key of the vehicle sender . Each message that is sent from one node to another contains many fields as shown below (Figure 4 and Figure 5):

Figure 4: Message format

Figure 5: message to RSU format

"$MessageType$" determines the type of message, e.g., authentication message, authentication request, or other type messages. "ID_S contains the real identity of the sender (or the PID_V pseudo-identity if the node is a vehicle). "ID_R" contains the receiver identity or the pseudo identity PID_V (if the node is a vehicle). "$(XPos, YPos)$" contain the coordinates of the location of the sender. "$Message$" contains an encrypted message ."$signature$" contains the signature of the "$Message$" field . "Pk_Sender" contains the public key of the sender. "$Timestamp$" contains the time at which the message was sent.

Algorithm 4 .Operations performed for the verification of a message

Input: received message M, time of acceptance of a message TA
Ouput:

$\triangle \leftarrow$ TA - *message*["*Timestamp*"];
Verif_PK $\leftarrow$ *verifier_PK*(*message*["*PK_Sender*"]);
signature_verif $\leftarrow$ *sign_verify*(*M*["*message*"],*M*["*signed_message*"], PK_Sender);
If verif_signature and verif_PK and $\triangle < TA$ **then**
  verify $\leftarrow$ True;
Else
  verify $\leftarrow$ False ;
End

4.3 Registration & authentication

- Registration phase

In this part, the CA generates a set of system parameters based on the unique UID provided by the registered vehicle. To preserve the confidentiality of the identity of the vehicle information and the communication, the CA generates a pseudonym for the registered vehicle for the communication. This pseudonym PID has a unique matching relationship with the real identity of the vehicle and is stored in the CA database. In addition, the CA also provides hash values to the registered vehicles to facilitate vehicle authentication and reduce the CA's communication load. In the registration process, the following steps are taken:

- Step 1: CA: firstly checks the existence of the real identity RID of the registered vehicle. If there is, then it generates the certificate that includes the pseudonym PID, a public-private RSA key pair (PK and SK), and the validity period VP, all of these elements corresponding to the real identity of the vehicle.
- Step 2: CA: computes two hash functions using th cryptographic hash function SHA256 [11], which are H0 = (PID ‖ PK_Vi) and H1 = (RID ‖ cert) in order to authenticate the vehicle's identity and store the vehicle information in the future. H0 and H1 will be sent to the Cloud Server and V respectively.
- Step 3: the Cloud Server stores the hash value H0 sent by the CA.
- Step 4: the registered vehicle V obtains a Pseudo ID PID, a certificate cert, H1, and a pair of public-private keys from the CA (PK_Vi, SK_Vi), and stores them in the on board unit (OBU).

The Vehicle V sends this certificate request periodically. Every VP (it's the certificate validity period), V sends a certificate request back to CA. In our system, CA designs two hash functions (H0 and H1). H0 obtains the pseudonym of the vehicle PID and the public key PK_Vi, which is stored in a cloud server. H1 represents the mapping relationship between the RID and the PID pseudonym and is stored in the OBU.
Only the CA and the registered vehicle V know the real identity RID of the vehicle. In addition, the relationship between the real identity RID and the assigned PID is

also stored in a hash map in the CA database. This ensures an easy search in case of tracking and verification of malicious vehicles. In addition, to reduce the dependency of the CA and prevent any malicious activity of the CA, H1 will also be stored on the blockchain at the vehicle authentication level. Thus, the CA and the vehicles will not be able to be repudiated.

- Authentication phase

Due to the high mobility of vehicles, a vehicle (V) may sometimes leave the area covered by an RSU U and enter an area covered by another RSU U'. In this case, V authenticates with the RSU U' and, therefore, changes the group. To avoid the possibility that no intruder can modify the list of vehicles in the RSUs, we propose the blockchain for the registration of vehicles in the RSUs. As we stated in Section 3.1, the data in a blockchain cannot be modified. RSUs as peer nodes build the blockchain network. Each RSU periodically (period TC) broadcasts its public key PK_R and a hash value of its identity Hash(ID_R) to all vehicles. When a vehicle V detects the signal from U', it may decide, according to a specific policy, to switch to U' (e.g., if the strength of the signal from U' is greater than that of the signal from U). V sends an authentication request with its own PID to the RSU. Furthermore, by calculating the result corresponding to the request, the RSU obtains the veracity of the result by making a query on the cloud server. Finally, the RSU stores the authentication result on the blockchain. During the authentication, vehicle V communicates with the RSU using its pseudonym. The RSU and the cloud server do not learn the real identity of the RID vehicle. This authentication process is divided into the following ten steps: Vehicle V launches an authentication request when receiving a broadcast message from RSU. V answers the RSU R by a message which contains the pseudonym PID and the public key PK_Vi to the neighboring RSU. This message is of the type "authentication request" and has the following structure (figure 6). This message must be hashed and signed by the vehicle private key SK_Vi, to prevent an adversary from tracking V by listening to its registration requests and to ensure that the message is not modified. Then, V puts this message under the structure (Figure 4).

– Step 1: Vehicle V launches an authentication request when receiving a broadcast message from RSU. V answers the RSU R by a message which contains the pseudonym PID and the public key PK_Vi to the neighboring RSU. This message is of type "authentication request" and has the following structure (figure 6). This message must be hashed and signed by the vehicle private key SK_Vi, to prevent an adversary from tracking V by listening to its registration requests and to ensure that the message is not modified. Then, V puts this message under the structure (Figure 4).

Figure 6: Authentication request

– Step 2: After receiving the request, the RSU R calculates the hash value according to the H0 algorithm and gets the corresponding result. To verify the legality of the vehicle, the RSU R requests the veracity of the result using the cloud server. If the query result is identical to the computation result, the cloud server returns TRUE to the RSU. Otherwise, the authentication fails.

– Step 3: After determining the authenticity of the vehicle identity, the RSU starts the process of negotiating the random integer SVi. And it computes H2 as the hash of its identity (H2 = hash (ID_ R)). Then, it sends to vehicle V a random integer SVi and H2 which are encrypted by the public key PK_Vi of the vehicle.

– Step 4: Vehicle V receives the message. First, it checks the message (signature and hash value). Then, V verifies the identity of RSU by comparing ID R, which is the broadcast message sender, and H2 (the hash value of ID R). Then, if the verification is true, V decrypts the ciphertext with its private key SK_Vi and then stores the integer SV in the OBU for later use in I2V communication . Vehicle V receives the message. First, it checks the message (signature and hash value). Then, V verifies the identity of RSU by the comparison between ID R, which is the broadcast message sender, and H2 (the hash value of ID R). Then, if the verification is true, V decrypts the ciphertext with its private key SK_Vi, then stores the integer SV in the OBU for later use in the I2V communication.

– Step 5: In order to determine if the vehicle V has received the integer from the RSU and the integrity of the random number, the vehicle must select another random number RVi, and then calculate a hash function H3 to obtain the two integers mentioned above (H3 = hash (SVi , RVi). In addition, in order to prevent malicious activity by the CA, the vehicle must sign H1 (i.e., the correspondence between the real identity and the certificate) with its private key SK_Vi and then store H1 in the blockchain network. The vehicle V prepares a message that contains the signature of H1 with its private key, H2 and the random value RVi: SignSk_Vi(H1) || H2(SVi || RVi) || RVi (as shown in figure 7). Then, it sends it to the RSU in the form of figure 5.

Figure 7: structure of message

– Step 6: After the verification of the received message(according to the algorithm 4) , the RSU uses PK_Vi to verify the hash value H1 with the vehicle signature and calculates the corresponding hash value based on RVi and the known negotiation random number SV. If the calculated result is the same as the received hash value H2, it means that the random number negotiation was successful.

– Step 7: the RSU, as a peer node of the blockchain network, prepares a transaction Tx which contains the signature of H1, the pseudo id of V, and the public key of V; Tx (sigSk_Vi(H1) || PID_Vi || PK_Vi) Then, it checks the existence of Tx in BC. If Tx does not exist in BC, then, U stores Tx in the blockchain network where all RSUs can obtain the transaction Tx. For authentication and identification of the source, each Tx transaction is digitally signed by the owner (i.e., RSU U) with his private key SK_R. And this transaction Tx put in a structure called "Block" uniquely identified by its hash and timestamp [2]. The validation of this transaction and the block is done by a consensus mechanism. This means that the state of the shared ledger is updated by the agreement of the majority of RSUs blockchain nodes. We choose to use the PBFT consensus algorithm (more details on PBFT in the section 3.2). Thus, the vehicle is registered in the RSU's BC. In the case where the transaction Tx exists in BC, i.e. V is already registered in BC. In this case, U checks the existence of the pseudo id of the vehicle and the public key PK_V in "the list of connected vehicles" LCV. If it is exist, then go to step 9.

– Step 8: Then, the RSU adds the vehicle pseudo id, the public key PK_Vi, and the time of the addition Taj in the list named "list of connected vehicles" LCV, and also stores the pseudo id of this vehicle with the random value SV and Taj in another list named "list of secret values" LSV.

– Step 9: RSU U sends an "authenticated message" to V. U prepares a message that contains the list of connected vehicles LCV . This message will be encrypted with the vehicle public key PK_Vi (this is the element "message" in Figure 4), and hashed and signed with the RSU private key SK_Vi. RSU sends this message under the structure of figure 4.

– Step 10: when receiving the "authenticated message" type message and after verification (according to algorithm 4), Vehicle V decrypts the message (message['message']) with its private key SK_V, then stores the list of connected vehicles under the name of "list of relay vehicles".

Since node mobility is high in the IoV, group membership can change frequently. To preserve the routing mechanism described in Section 4.1, group members must be informed of group changes. In our VIPER enhancement, this task is accomplished by the vehicle V, by periodically sending authentication messages. This sending updates the list of connected vehicles in R. Consequently, it also updates the list of vehicles in V.

4.4 RSU replies

There are some messages (mainly Location-Based Service requests) that require a response from the RSU. For example, a routing request to a particular destination sent by a vehicle V to the RSU via an LBS request requires a response containing the route itself. In these cases, the privacy of vehicle V must be protected not only at the phase of sending the request but also at the phase of transmitting the response. First, the RSU cannot broadcast the response message using a plain-text destination field, because an attacker could follow the

vehicle route by listening to the RSU's responses to vehicle V [4]. Moreover, it is impossible to broadcast the encrypted message with the public key of V, because each vehicle would be trying to decrypt this message to find out whether it is the intended recipient or not. It is noted that this could introduce unnecessary overhead since the decryption of the public key is a computationally expensive operation. The solution to this problem is to use the HMAC algorithm to enable a vehicle to discriminate whether it is the intended recipient of the message without requiring the decryption of the message. The RSU and V share the secret value SV (they exchange it secretly during the registration phase - section 4.3). Each response from the RSU to the vehicle is then encoded with the public key of the vehicle and broadcast with a random value r and the HMAC calculated on the random value r and the secret value SV - i.e., HMAC(SV;r). Once a vehicle receives the triplet: random value, HMAC, and encrypted message (< r; hmac; enc_message >), it calculates the value hmac' = HMAC(r; SV). If hmac'= hmac , then it is the expected recipient of the encrypted message and can perform the decryption of the message. This operation is computationally expensive since it is not the expected recipient [4].

5.EVALUATION

5.1 Security Analysis

– PROTECTION AGAINST TRAFFIC ANALYSIS ATTACK

To verify that our protocol is resistant to traffic analysis attacks, three attacks have been considered: the message encoding attack, the message volume attack, and the timing attack. In the case of a message coding attack, if messages do not change their coding during transmission, they can be linked or traced. This proposed approach is resilient to this type of attack due to the third step of the algorithm 2 at the relay vehicle. In case the next receiver is the relay vehicle, the sender concatenates the encrypted message (encrypt_message_R) with the pseudo-identity of the relay vehicle (the next receiver), and then signs it. Otherwise, the vehicle keeps the encrypted message (encrypted_message_R) and signs it. Since the relay vehicle performs these steps, the encoding of the message changes with each relay, which makes it impossible for the adversary to track the message. In the second type of attack "message volume attack", an adversary of this type of attack can observe the duration of a specific communication by connecting its possible endpoints and expecting a correlation between the creation and/or release event at each possible endpoint. The VIPER protocol and our proposal are resistant to this type of attack because the size of the message and the batch are fixed.

– PRIVACY PRESERVATION

The vehicle Vi uses its own pseudonym PIDi issued by CA to perform V2V and V2I communications without any information about its real identity RID. To find a compromise between security and privacy, the identity and pseudonym pairs are stored with a high level of security in the CA. In other words, only the CA knows the real identity of all pseudonyms issued for each vehicle. Thus, only the CA has the power to track the malicious vehicle when it misbehaves or broadcasts false messages. In addition, the link between the real identity and the pseudonym is also stored on the blockchain at the authentication level of a vehicle. The RSUs cannot know the link-specific information, which effectively improves the trust level of the CA. In order to preserve the privacy of vehicles, the transaction Tx does not contain any information that can be linked to the real identity.

– INTEGRITY

This requirement is guaranteed by cryptography, since all exchange in this proposed protocol is based on certificates (long term, short term) and public keys, all exchange is encrypted and the data is protected.

– PROTECTION AGAINST TRANSACTION TAMPERING

In this proposal, the transactions recorded on the blockchain have already obtained the agreement of RSUs, and the blockchain maintains the interactive consistency of RSUs. A hash chain is used to guarantee the order and information of blocks. These hash values are unique for each block. Changing the contents of any block will cause the hash values of the other blocks to change. According to the properties of the hash function, if a malicious adversary or RSU starts a perfect forgery, it must not only change the contents of the block, but also change and recalculate the hash values of all blocks after the changed block. Therefore,

when there are hundreds of blocks, regardless of the workload, the consensus adopted in the blockchain composed of all RSUs is the practical PBFT Byzantine fault tolerance. So, as long as the malicious node does not exceed half of all RSUs, the transaction information cannot be changed, so the longer the blockchain, the higher the security will be.

– PROTECTION AGAINST SYBIL ATTACK & MESSAGE FABRICATION ATTACK

For the Sybil attack, a malicious node uses multiple identities at the same time. In the proposed approach, a malicious node cannot use multiple identities because each node only gets one pseudo-identity and two keys (public and private key) when its certificate is verified in the registration process. For message generation, a malicious node tries to modify or delete the message. In the proposed improvement, when a vehicle sends a message, it generates a digital signature.This digital signature is computed by hashing the entire message and encrypting it using the private key. When an RSU or vehicle receives the message, it decrypts it with the sender's public key and verifies its integrity.

5.2 Performance Analysis

5.2.1 Simulation

The simulation is carried out by using NS-3.35 simulator [21], with simulation parameters illustrated in Table 2. We used NS 3.35 for its robustness and maturity, which is the industry standard for simulations, experiments and testing in other studies [21]. The simulation is done in a personal computer using a virtualization process (VMware). Linux Ubuntu(16.04) is the operating system. The processor is Intel Core i7 with assigned 4 GB RAM. Our experiments were conducted using two different scenarios as follows: In the first scenario, we used four RSUs and 1 to 80 vehicles. This scenario uses to examine the computation overhead in the registration and authentication process. The second scenario is the same as in [4], made up of a square area entirely covered by one RSU (all the vehicles in the simulation belong to the same group).

Table 2: SIMULATION PARAMETERS

Parameter	Value
Simulation area	1000m * 1000m
Scenario simulation time	80 min
Number of vehicles	1 - 80
Asymmetric key cryptography	CRT-RSA

5.2.2 Experiments

We measure the time cost for the registration and authentication part of the proposal to present performance of this scheme. As shown in Figure 8, when a vehicle is registered with CA, there are mainly two hash functions to be computed (H0 and H1), the average time to register a vehicle is about 0.01.

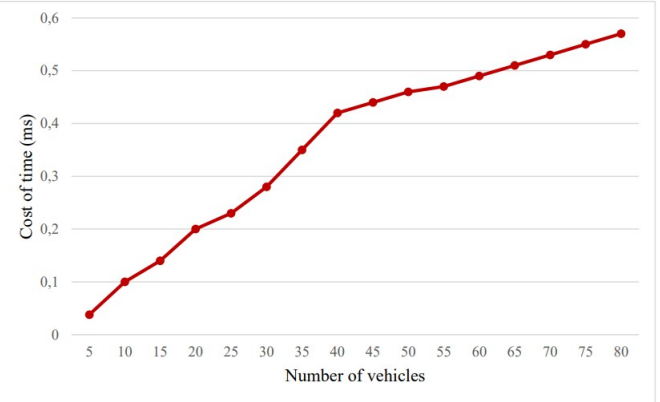

Figure 8: registration time.

Figure 9 shows a comparison of the transaction confirmation time for a block of data in our proposal using PBFT and using POW (the most well-known consensus algorithm). As shown in this figure, the transaction confirmation time for a data block in our proposal has been set to 0.012 s, while the transaction confirmation time for a block of transactions for the traditional blockchain (i.e., BC with POW) is 0.031 s.

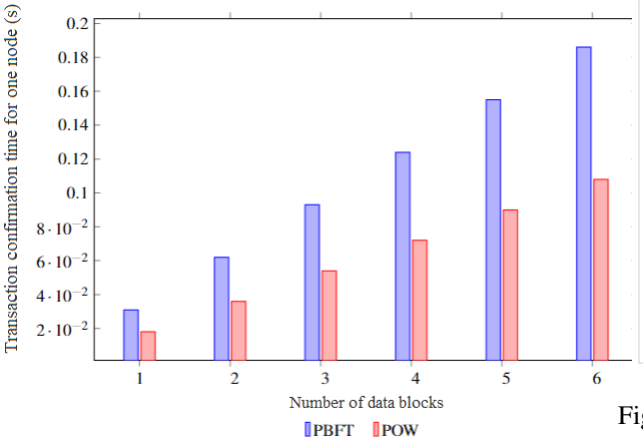

Figure 9: Comparison of transmission performance

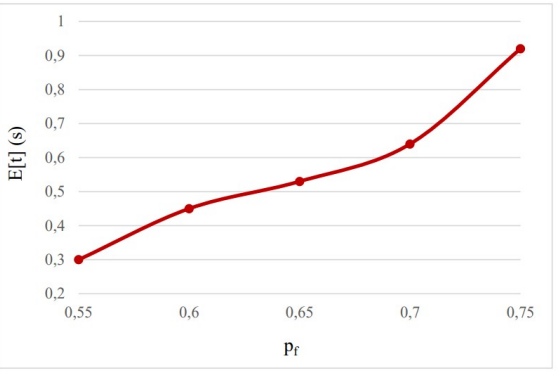

Figure 10: Expected value of the message delivery time.

We repeated the second scenario in 100 iterations to observe the following results. This protocol was analyzed based on two parameters, namely the communication rate between vehicles, and the message delivery time as a function of the message transmission probability pf. We note that we use the same parameters as used in [4]. In particular, we use equation (1), which is used to calculate the message delivery time. In this protocol, the sender sends a batch of messages at the end of a time slot and each relay vehicle waits for a time slot and then forwards the messages, to another relay vehicle or the RSU. The expected value of the message delivery time can be calculated by equation (1), where t_{ts} is the length of the time slot. E [t] as a function of pf is shown in Figure 10 , using the same value of t_{ts} as in [4], which is 300 ms.

$$E[t] = \frac{t_{ts} \times p_f}{1 - p_f} \quad (1)$$

The value 0.6 was chosen for pf in [4]. Thus, according to figure 10, the value 0.6 of pf determines an expected delivery time of 0.45 s. Therefore, we notice that this result is the same in the VIPER. To be more precise, we collected Figure 11 during the simulation of our proposal and VIPER protocol.

In Figure 11, the message delivery time as a function of pf is plotted. The simulation results confirm that the choice of pf = 0.6 corresponds to an average message delivery time of 0.45 s. This delay is suitable for applications without time constraints. The curve trend of our proposal is similar to that of the VIPER protocol (Figure 11). This means that our proposal maintains the same message delivery delay as the VIPER protocol. Figure 12 shows the analysis of throughput versus vehicle number.

5.2.3 Overhead Analysis

We evaluated the computation overhead effect in our protocol and compared the effects on the total time needed to complete this phase to the results from [4]. The experiments are performed with three different scenarios. In the first scenario, we used the algorithm described in [4] in our approach. It is the universal re-encryption ELGAMAL algorithm [8]. Second scenario modifies the previous scenario by using CRT-RSA as public key cryptography algorithm. Finally, the third scenario is performed with the traditional RSA cryptography. The results of the cryptography operations in these scenarios were obtained by the average of 1000 measurements and for three different key sizes (512-bit, 1024-bit and 2048-bit). The simulation results of the three scenarios are illustrated in Table 3 and 4 , Figure 13 and 14.
Table 3 and Figure 13 represent the analysis of the encryption time with different algorithms, i.e., the time required to encrypt the data. This encryption operation is performed at the OBU level.

The results show that the first scenario has the highest average computation delay. The average calculation time in the first scenario is reduced in the two other scenarios. In the second scenario, it is reduced by 50%, as well as in the third scenario. As mentioned in [4], the CRT-RSA algorithm has not modified the encryption

phase of RSA, which means that the encryption time of CRT-RSA is the same as RSA.

Table 3: Encryption time (ms) - at the OBU

Scenarios	**key size (bits)**		
	512	1024	2048
With Universal Re-Encryption ELGAMAL	0.02 ms	0.71 ms	1.82 ms
With Traditional RSA	0.132 ms	0.35 ms	0.912 ms
With CRT-RSA (the used algorithm)	0.132 ms	0.35 ms	0.912 ms

Then, we analyze the decryption time at the RSU, as shown in Table 4 and Figure 14. The decryption computation time results differ for RSA, RSA-CRT, and universal re-encryption ELGAMAL algorithms. Universal re-encryption ELGAMAL requires the most time to decrypt compared to RSA. On the other hand, the advanced version of RSA named CRT-RSA produced the highest decryption gain among all techniques. Indeed, the test results show that the decryption of CRT-RSA algorithm, which is used in our proposed protocol, is faster and more efficient than traditional RSA and ELGAMAL universal re-encryption in the decryption process.

Table 4: Decryption time (ms) - at the RSU

Scenarios	**key size (bits)**		
	512	1024	2048
With Universal Re-Encryption ELGAMAL	0.02 ms	0.71 ms	1.82 ms
With Traditional RSA	0.931 ms	1.46 ms	2.76
With CRT-RSA (the used algorithm)	0.391 ms	0.607 ms	1.47 ms

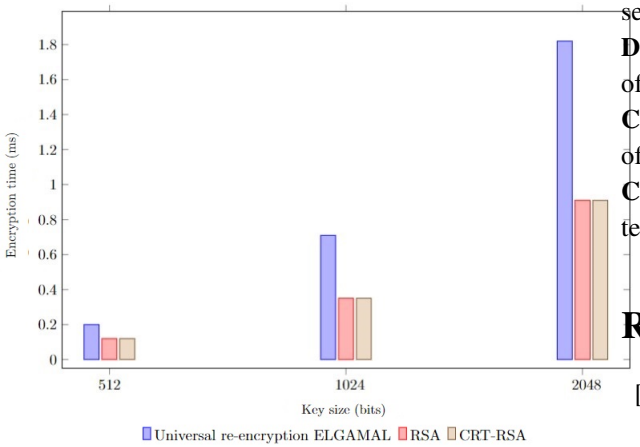

Figure 11: Encryption time.

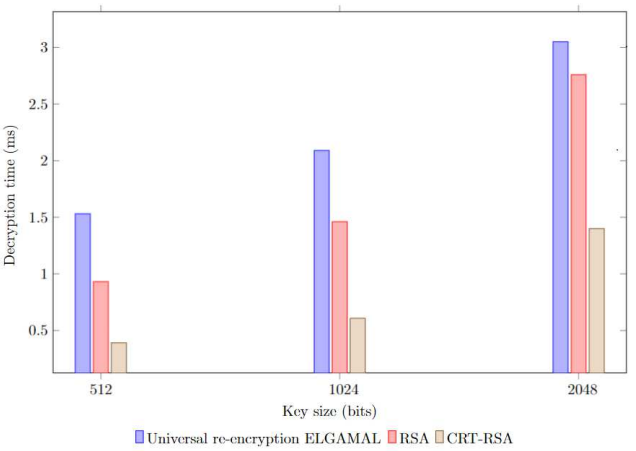

Figure 12: Decryption time.

6.Conclusion

The increasing number of smart vehicles creates new attack platforms. Therefore, there is an appropriate need for security in the IoV scenario. In this paper, we designed a patch on the Cencioni and Pietro protocol [4] using blockchain technology and certificate authority. The analysis proved that the patched protocol was more secure and efficient than the original protocol.

Supplementary Materials and declarations

The supplementary materials used to support the findings of this study are included within the article.

Funding The authors declare that the present research is not supported by any one.

Data Availability The data used to support the findings of this study are included within the article.

Code Availability The code used to support the findings of this study are inclided in the article.

Confict of interest The authors declare no confict of interest.

References

[1] Hilmi Abdullah and Ali Hikmat Ibrahim. Blockchain technology opportunities in Kurdistan, applications and challenges. *Indonesian Journal of Electrical Engineering and Computer Science*, 18(1):405, April 2020.

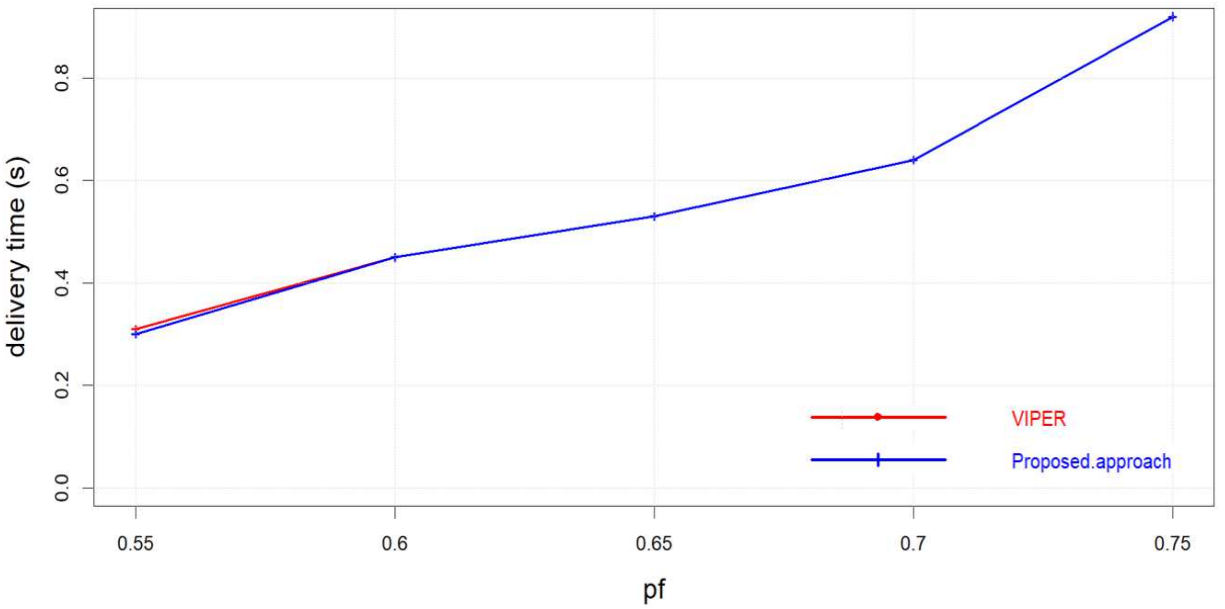

Figure 13: Message delivery time.

[2] Robert Anascavage and Nathan Davis. Blockchain Technology: A Literature Review. *Blockchain Technology*, page 10, 2019.

[3] Miguel Castro and Barbara Liskov. Practical byzantine fault tolerance and proactive recovery. *ACM Transactions on Computer Systems*, 20(4):398–461, November 2002.

[4] Paolo Cencioni and Roberto Di Pietro. A mechanism to enforce privacy in vehicle-to-infrastructure communication. *Computer Communications*, 31(12):2790–2802, July 2008.

[5] David L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM 1981-feb 01 vol. 24 iss. 2*, 24, feb 1981.

[6] Debashis Das, Sourav Banerjee, and Utpal Biswas. A secure vehicle theft detection framework using Blockchain and smart contract. *Peer-to-Peer Networking and Applications*, 14(2):672–686, March 2021.

[7] Xia Feng and Jin Tang. Obfuscated RSUs Vector Based Signature Scheme for Detecting Conspiracy Sybil Attack in VANETs. *Mobile Information Systems*, 2017:1–11, 2017.

[8] Philippe Golle, Markus Jakobsson, Ari Juels, and Paul Syverson. Universal Re-encryption for Mixnets. In Gerhard Goos, Juris Hartmanis, Jan van Leeuwen, and Tatsuaki Okamoto, editors, *Topics in Cryptology ? CT-RSA 2004*, volume 2964, pages 163–178. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004. Series Title: Lecture Notes in Computer Science.

[9] Qinglei Kong, Rongxing Lu, Maode Ma, and Haiyong Bao. A privacy-preserving sensory data sharing scheme in Internet of Vehicles. *Future Generation Computer Systems*, 92:644–655, March 2019.

[10] Haiqing Liu, Yan Zhang, Shiqiang Zheng, and Yuancheng Li. Electric Vehicle Power Trading Mechanism Based on Blockchain and Smart Contract in V2G Network. *IEEE Access*, 7:160546–160558, 2019.

[11] Ali Maetouq, Salwani Mohd, Noor Azurati, Nurazean Maarop, Nilam Nur, and Hafiza Abas. Comparison of Hash Function Algorithms Against Attacks: A Review. *International Journal of Advanced Computer Science and Applications*, 9(8), 2018.

[12] Leandros Maglaras, Ali Al-Bayatti, Ying He, Isabel Wagner, and Helge Janicke. Social Internet of

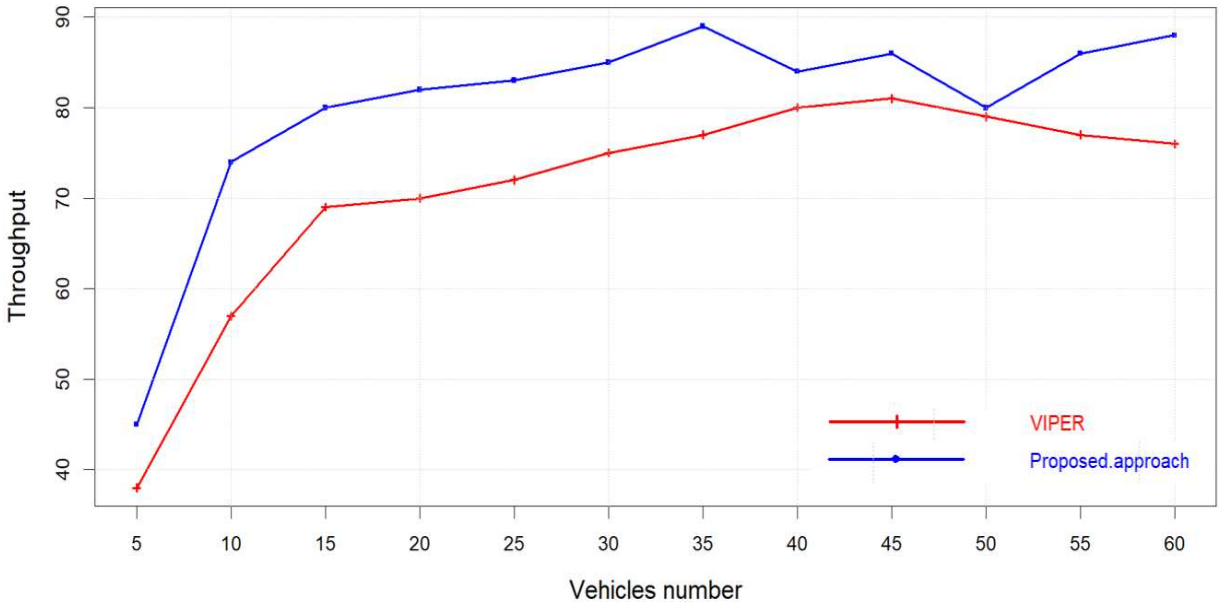

Figure 14: Analysis on Throughput.

Vehicles for Smart Cities. *Journal of Sensor and Actuator Networks*, 5(1):3, February 2016.

[13] Zaigham Mahmood, editor. *Connected Vehicles in the Internet of Things: Concepts, Technologies and Frameworks for the IoV*. Springer International Publishing, 2020.

[14] Ming-Chin Chuang and Jeng-Farn Lee. TEAM: Trust-Extended Authentication Mechanism for Vehicular Ad Hoc Networks. *IEEE Systems Journal*, 8(3):749–758, September 2014.

[15] Shiva Raj Pokhrel and Jinho Choi. Federated Learning With Blockchain for Autonomous Vehicles: Analysis and Design Challenges. *IEEE Transactions on Communications*, 68(8):4734–4746, August 2020.

[16] Danda B. Rawat, Ronald Doku, Abdulhamid Adebayo, Chandra Bajracharya, and Charles Kamhoua. Blockchain Enabled Named Data Networking for Secure Vehicle-to-Everything Communications. *IEEE Network*, 34(5):185–189, September 2020.

[17] Ashish Sharma and Dinesh Bhuriya. Literature Review of Blockchain Technology. 6(1):8, March 2019.

[18] G N Shinde and H S Fadewar. Faster RSA Algorithm for Decryption Using Chinese Remainder Theorem. page 8, 2008.

[19] Lama Sleem, Hassan N. Noura, and Raphaël Couturier. Towards a secure ITS: Overview, challenges and solutions. *Journal of Information Security and Applications*, 55:102637, December 2020.

[20] Victor; Jegatha Deborah L.; Balusamy Balamurugan; Shynu P.G. Vijayakumar, P.; Chang. Computationally efficient privacy preserving anonymous mutual and batch authentication schemes for vehicular ad hoc networks. *Future Generation Computer Systems 2018-jan vol. 78*, 78, jan 2018.

[21] E. Weingartner, H. vom Lehn, and K. Wehrle. A Performance Comparison of Recent Network Simulators. In *2009 IEEE International Conference on Communications*, pages 1–5, Dresden, Germany, June 2009. IEEE.

[22] Yuan Yao, Bin Xiao, Gaofei Wu, Xue Liu, Zhiwen Yu, Kailong Zhang, and Xingshe Zhou. Multi-Channel Based Sybil Attack Detection in Vehicular Ad Hoc Networks Using RSSI. *IEEE Transactions on Mobile Computing*, 18(2):362–375, February 2019.

[23] Zhaohui Zhang, Sanyang Liu, Yiguang Bai, and Yalin Zheng. M optimal routes hops strategy:

detecting sinkhole attacks in wireless sensor networks. *Cluster Computing*, 22(S3):7677–7685, May 2019.