Received: 28 April 2022 | Revised: 9 August 2023 | Accepted: 24 November 2023

DOI: 10.1002/cpe.7981

RESEARCH ARTICLE

WILEY

A fog-driven three-factor authentication protocol for secure data sharing in Internet of Vehicles cyber-physical systems

Siddharth Katyal | **Shashank Gupta** | **Oshin Rawlley** | **Debjani Ghosh**

Department of Computer Science and Information Systems, Birla Institute of Technology and Science, Pilani, Rajasthan, India

Correspondence
Shashank Gupta, Department of Computer Science and Information Systems, Birla Institute of Technology and Science, Pilani, Rajasthan, India.
Email: shashank.gupta@pilani.bits-pilani.ac.in

Funding information
CHANAKYA Fellowships of IITI DRISHTI CPS Foundation under the National Mission on Interdisciplinary Cyber Physical System (NM-ICPS) of Department of Science and Technology, Government of India, Grant/Award Number: DRISHTI CPS/SL/CF/22-23/04

Summary

The Internet of Vehicles (IoV) has potentially escalated the management of vehicle and route planning. However, as the IoV becomes more prevalent, safeguarding the privacy and security of IoV data becomes crucial. Additionally, because vehicles often operate with minimal human interference, they become susceptible to various types of attacks that can compromise their privacy and security. In this article, we propose a three-factor authentication system based on CROPUF (Crossover Ring Oscillator Physically Unclonable Function) that not only secures transmissions but also ensures safety against physical intervention. The three-factor authentication protocol (passwords, biometrics, and PUF) avoids registering any confidential information in the user device eliminating the possibility of intrusion by an intruder. We also propose the use of aggregators (charging stations that act as mediators between the IoV and the data center) as a fog layer communication which will reduce load and improve the efficiency of communication between vehicles and the data center. Finally, the security analysis validates that our proposed method can survive several well-known assaults and achieve the required features of security by maintaining the user anonymity.

KEYWORDS

aggregators, CROPUF, fog layer communication, Internet of Things, Internet of Vehicles, PUF, three-factor authentication

1 | INTRODUCTION

1.1 | Background

An IoT ecosystem involves large amount of data processing and transmission such as real-time traffic data, vehicle mobility information, and so forth.[1] With the advent of intelligent transportation systems (ITS), Vehicular Ad Hoc Networks (VANETs) have attracted substantial amount of consideration as a wireless communication mechanism facilitating mobile networks for inter-vehicular communication. Moreover, the vehicles are equipped with communication equipment supporting wireless transmission and as these transmissions takes place Internet of Vehicles (IoV) has emerged as a novel research area for building ITS.[2] The communication technologies such as dedicated short-range communication (DSRC) is utilized for volatile vehicles and for vehicle to infrastructure (V2I) communication. As the communication takes place inevitably between the vehicle's security concerns rise because data transmission takes place openly on wireless medium. Therefore, it is important to develop and design an authentication protocol specially for V2I and V2V communication for ensuring the security of driver.[3] Furthermore, for building secure and safe authentication protocol, there exists many challenges as follows such as:

- The area comprising of many vehicles should be partitioned so that the communication overhead reduces.
- Second, the session key agreement structure should entail in the inter-geographical regions for the ensuring a seamless communication between vehicles.

Nowadays, authentication protocols are persistently designed to cater the above-mentioned issues. Hence, in this article we develop and design a lightweight protocol for IoV communication by combining CROPUF, biometrics, and password. The system model of the proposed protocol is designed for encouraging inter-region communications.

The exponential rise in the number of vehicles has led to a corresponding surge in challenges that are associated with controlling the traffic, incidents of road accidents, and optimal route allocation. In the IoV, vehicles equipped with an amalgamation of various IoT devices (such as sensors) are connected over a huge network to share individual information resulting in efficient vehicle management and improvement of state of roads. IoV evolved from VANET and is expected to ultimately grow into an Internet of autonomous vehicles in the future. It is a new area in the automobile business that is becoming increasingly significant in smart cities. Figure 1 shows the evolution of IoVs over the past couple of decades.

The data is collected from a specific environment by various embedded devices in the vehicles. IoV aims to improve the users' travel time and safety thereby improving the overall experience by utilizing the collected data to manage resources. Automobiles, data hub, roadside units (RSU), and people are all part of the IoV system. As a result, in IoV, information may be exchanged in the form vehicle-to-vehicle (V2V) communication, vehicle-to-infrastructure (V2I), vehicle-to-people (V2P), and vehicle-to-onboard-sensors (V2S).[4]

The main motivation of this article is addressing security concerns in vehicular communications. Apart from the conventional security measures, the anonymity and privacy of vehicular users have emerged as critical factors to be considered when designing communication protocols. Communicating the identity of vehicular users in plaintext can lead to many issues, allowing adversaries to potentially track the current location of the users, their movement, login history, and so forth, especially in mobile communication scenarios. Further, to safeguard communication privacy in vehicles, security experts have developed several security protocols. However, these protocols overlook the possibility of physical attacks on vehicles and fail to support authentication of seamless handover when a vehicle moves between RSU (Roadside Unit) coverage areas.

During vehicle-to-DC communication, an aggregator, which also serves as a charging station, acts as an intermediary or gateway, thereby, functioning as a fog layer. It reduces the load on DC, enhances communication frequency, and reduces response time from users to DCs. In such circumstances, the communication channels between the user to RSU or aggregator and between aggregator to DCs both need to be secure. The vehicles are supposed to upload data related to the vehicle and its position such as its speed, fuel consumption, road conditions, and so forth. The data is then communicated to the DCs where its compiled and analyzed and converted into a meaningful resource. Figure 2 shows the role of vehicles, fog layer, and data center in a typical IoV system.

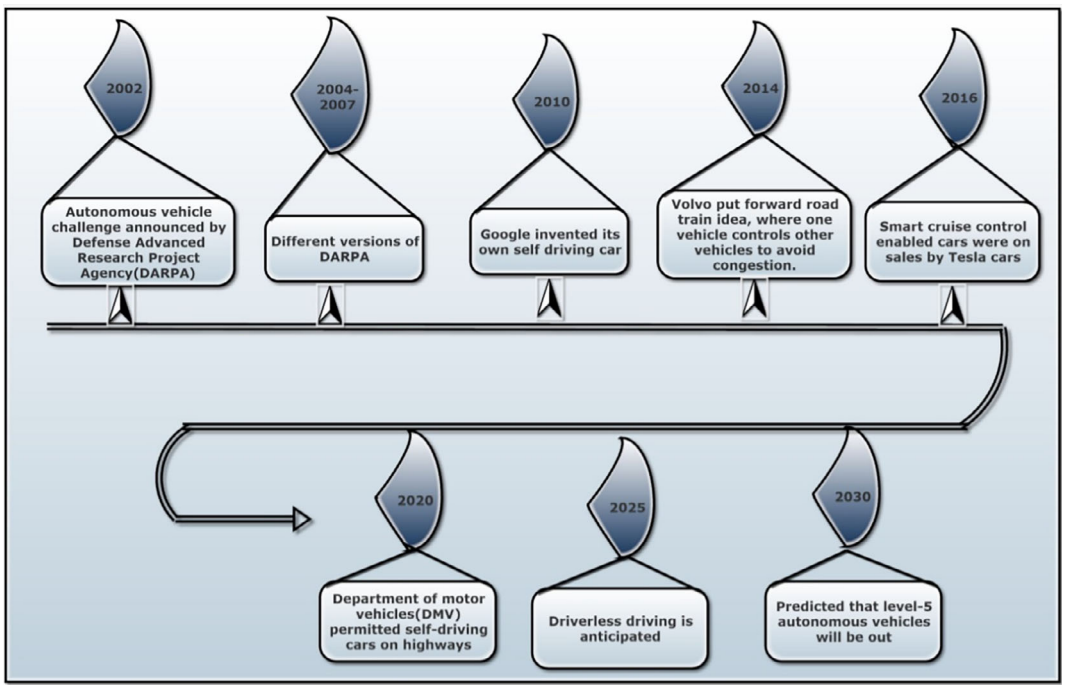

FIGURE 1 Evolution of IoVs over time (from VANETS to autonomous vehicles).

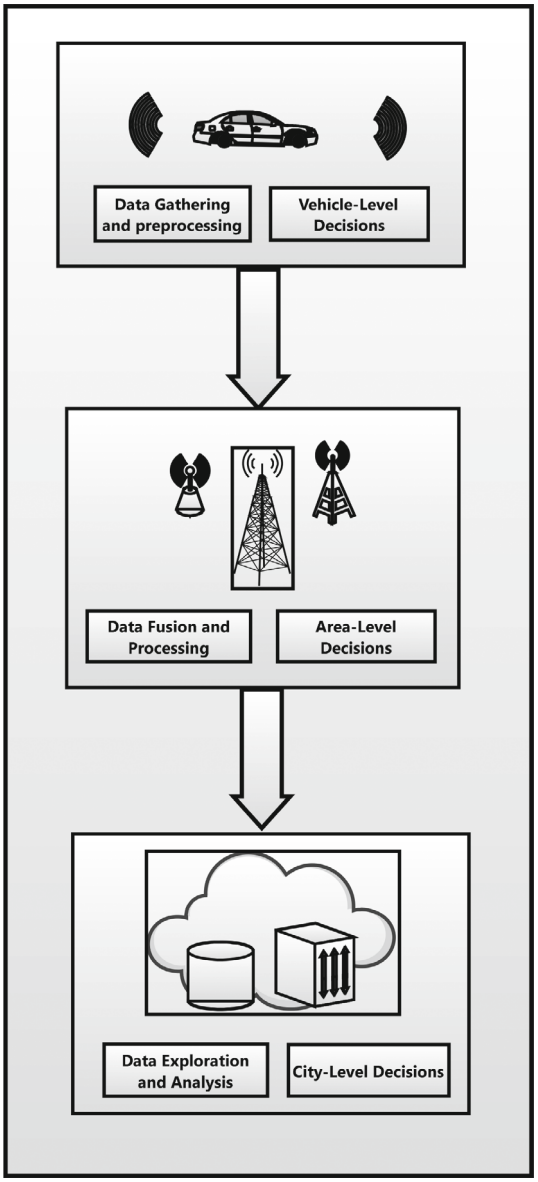

FIGURE 2 Role of various parts of IoV system (fog-layer communication).

The data usually contains private information about the driver and vehicle, such as its real-time position, make and model, and so forth. However, because data communication channels are public, an attacker may simply acquire sensitive data, jeopardizing data privacy.[5] Furthermore, the adversary may inject bogus location data and information about the accident, thereby jeopardizing the security of IoV ecosystem.

In the earlier times, to guarantee privacy and security, electronic devices were obligated to store confidential data in either electrically erasable programmable read-only memory (EEPROM) or battery-backed non-volatile static random-access memory (SRAM). In the modern era, cryptographic algorithms were employed to encrypt the information and ensure authentication. However, most IoT devices have limited resources in terms of CPU, memory, battery, and so forth, hence, they cannot support traditional methods of security. In addition, the vehicles are usually left alone with little to no interaction and hence, an adversary can take advantage of this situation making vehicles prone to physical attacks.

Thus, to ensure the privacy of users and the security of vehicles, an authenticated key exchange (AKE) protocol is necessary, which safeguards from eavesdropping attacks and physical harm, that is, even if the sensors of vehicles are compromised the security and anonymity should not be at risk. For device authentication and key generation, the silicon-based physical unclonable function (PUF) was developed as a novel hardware primitive that offers a unique device-dependent mapping from challenges to responses based on the unclonable features of the underlying physical properties of the device.[6] Since the underlying nanoscale structural disorder would be wrecked during physical tampering, the PUF generated key can survive tampering attacks.[7] Recently PUFs have garnered a lot of interest in academia and business as they can achieve authentication and

encryption/decryption with minimal design and structure.[8–11] Current weak PUFs, on the other hand, have a flaw when utilized in particular security protocols. They create a chip-unique key for each device that cannot be copied owing to process variance, but certain security protocols, such as multiparty communication, need multiple parties to share the same key. As a result, existing weak PUFs are ineffective in such circumstances.

The existing techniques have failed to guarantee privacy of the user allowing the attackers to easily sniff on the users.[12–16] However, the protocol in Reference 17 claims to ensure user anonymity, it is unsuccessful in accounting for the fact that information can be lost during transmission, and hence it is vulnerable to desynchronization attacks. Furthermore, the protocol in Reference 18 ignores the fact that a PUF's output is not perfect, and it contains some noise, and the protocol can even reject a legitimate user due to this. As a result, developing a resilient protocol based on PUF namely: AKE, is a difficult challenge. In this article, a three-factor authentication protocol using CROPUF is designed for the IoV system. The proposed scheme not only ensures the security of communication from vehicles to data centers but also guarantees the secure communications that occur between the aggregator and data center. The use of the fog layer reduces computation time and improves the overall communication experience for both the user as well as the operators. The use of CROPUF facilitates shared key generation with minimal effort and resources. The major contributions of the article are as follows:

- We propose a secure, lightweight scheme for IoV communication where all the channels between vehicles to the aggregator and aggregator to the data center are secure and help in maintaining user anonymity.
- We propose a three-factor authentication protocol by combining CROPUF, biometrics, and password, an adversary that will be unable to tamper with the device without verifying all three factors and hence, proving effective against eavesdropping as well as physical tampering. The proposed algorithm fuses the biometrics with the PUF response increasing authentication accuracy hence making it difficult to counterfeit assaults as it is difficult to counterfeit multiple features at once. Fuzzy commitment is also used to account for the noise in the PUF response and biometrics.
- The proposed technique allows for the cost-effective (with minimal effort and resources) creation of shared keys, the shared keys generated are then used as session keys to encrypt and decrypt the confidential information which would be shared eventually.
- The proposed scheme makes use of aggregators as fog layer systems that serves the dual purpose of acting as charging points as well as improving the communication by addressing the latency issues and making the communication between data centers and vehicles faster.
- Finally, we offer a comprehensive security analysis that demonstrates that the proposed protocol can survive several well-known assaults. Performance evaluation in terms of communication and computation overhead is conducted.

The organization of this article is explained as follows. Section 2 reviews the recent state-of-the-art regarding the PUF authentication protocols. Section 3 provides preliminary information about the PUF-based authentication protocols. The proposed system model of three-factor authentication based on CROPUF is discussed in Section 4. Formal and informal security analysis is conducted in Section 5. Section 6 presents a comparative analysis of proposed protocol with related protocols. Section 7 evaluates the performance of the proposed protocol based on two parameters. Section 8 conducts a simulation on NS-3 followed by the result evaluation in Section 9. Section 10 discusses and summarizes the existing and present state-of-the-art works in reference to our proposed work.

2 | PRIOR WORK

2.1 | Recent state-of-the-art

PUFs are classified category of device where the manufacturing process introducing the interior variations are exploited by PUFs. The PUFs taking input is termed as a "challenge" and the output given is termed as a "response." Many researches have been conducted on embedded security in vehicles. Researches such as identification of the components, protection of the vehicular software and in-vehicle secure communications has been focused more.[19,20] However, we observe that the attack types in the recent years have also changed owing to the change in the manufacturing, make, novel system of the vehicles. Scheibel et al.[21] suggested a vehicular S/W protection architecture which renders the software to be bound to a particular hardware of a vehicle and conforms to its software configurations. This ensures that the accessibility of the content of the content provider is limited only to the approved vehicles having suitable decryption secret key. Adelsbach et al.[22] recommended a protocol which enabled a secure installation and distribution of the SW in an embedded system. It uses an encryption system based on public key broadcast and reliable computation to make the S/W conform to a particular embedded system. Wolf[23] proposed numerous measures on the basis of present modern cryptographic methods for different vehicular security issues namely secrecy, authentication, and so forth, and also emphasizes the relevance of communication security within a vehicle which will enable the forthcoming services in the locomotive industry. Bogdanov et al.[24] gives solutions for security of architectures on the basis of hardware modules in a survey form in the domain of automotive applications.

Numerous protocols in the recent state-of-the-art are proposed for the charging system of the electric vehicles (EV) for allowing an authenticated key agreement amongst vehicles and charging system. This ensures a secure mutual authentication between them. Present protocols[25–36]

of authentication utilizes various primitives of cryptography such as symmetric/asymmetric key cryptography and blockchain technology. Li et al.[25] has proposed a fast authentication protocol to dynamically charge the EVs. The charging system in this protocol plays an integral role for providing secure communication between RSUs and EVs. A verification process holds before proceeding further between RSUs and EVs. Li et al.[26] have opted for pre-key-distribution method for generating and distributing keys using symmetric key cryptography amongst EVs, charging systems and other entities. Conditional privacy is facilitated by this protocol for EVs. Rabieh and Wei[27] devised a privacy-aware protocol that mutually authenticates the EVs and the charging pads without the involvement of a third entity. The privacy of the driver is conserved even while charging process. However, they have not considered many other security attacks that may impact the whole system.

Gunukula et al.[28] introduced secure communication-protocol acting between the EVs and the charging system and protected the vehicle by generating coins utilizing the partially blind signatures and thereby, making the vehicle difficult to be found out. In accordance with our analysis of the recent protocols, the state-of-the-art lacks security requirements and on the other hand some protocols compromise performances. Consequently, we propose a robust and safe ML attacks resilient protocol using PUFs.

2.2 | Configurable PUFs

Maiti and Schaumont[37] proposed a configurable RO PUF to increase the dependability of RO PUF. The important concept is that at each stage of the RO, a 2 1 multiplexer is utilized to pick one of two inverters. To enhance the reliability of PUF, this strategy uses the configurations with the highest delay difference. In Reference 38, another highly customizable RO PUF was presented. The essential concept is to employ a 2:1 multiplexer to choose or skip the inverter to increase the dependability of RO based PUFs.

2.3 | PUF based authentication protocols

The Physically Unclonable Function (PUF) is an innovative security tool having immense potential. Chatterjee et al.[39] utilized PUF to create a unique public identification key of devices. It makes identity-based privacy possible for IoT tools. For node-to-node communication and node-to-server communication, many authentication methods were suggested relying on PUF. Xie et al.[40] devised an authentication protocol based on the principles of Physically Unclonable Function intended for body area networks well suited for minimal memory IoT devices. Baruah and Dhal[41] developed a privacy model for home automation by integrating the one-time password (OTP) verification technique with PUF. Harishma et al.[15] created an AKE system to combine privacy and efficiency by delegating complex encryption algorithms on servers without memory constraints.

The PUF output depends upon the environmental conditions such as temperature change and therefore, noise must be taken into consideration, nevertheless, most of the three-factor papers have overlooked this factor. In some papers even the user's private key, that is, identification is shared in plain text. While others store all the responses to a challenge just to verify one single PUF device. Fuzzy extractors can solve the problem of noise. A combination of BCH (Bose Chaudhuri Hocquenghem) code and fuzzy commitment can be used to deal with the problem effectively. Making the use of PUF provides versatility and also saves storage as the device needs to store nothing more than PUF related information, this also provides security in case the device gets stolen. In Reference 17 the authors presented the idea of combining PUF and AKE to form a two-factor authentication algorithm. By integrating PUF, IBE, and hash-based encryption, Chatterjee et al.[39] created an AKE protocol. It made use of BCH coding to achieve noise correction for PUF. In Choi et al.[42] presented an idea of a two-factor authentication system that combines PUF response along with the physical environment recorded by the device, this prevents physical theft of the device as the environment will change as soon as it is stolen. They also used fuzzy commitment to combine the two factors.

In some of the above-discussed papers, the shared server identity which is sent in the plain text format is vulnerable to eavesdropping. Moreover, most of them did not consider the scenario when a transmission failed and hence the information could not be synchronized across devices as in the case of Reference 43. In Reference 43 the user ID and CRP are updated while verifying a user hence, if it fails to update, the user will not be able to log in to device ever again. Reference 17 also had the problem of physical theft as the device itself stores some identity verification variables which help the user to log in to the device.

2.4 | PUF for shared key generation

All nodes in a multiparty communication system must use the same keys to encode and decode the messages sent across the communication channels. Weak PUFs have a small set of CRPs and they might end up in the generation of the same key to be distributed over different communication channels and hence, rendering the entire process as useless. While, the resilient PUFs have a huge set of CRPs that behave the same as one-way hashes. Hence it is almost impossible to generate a shared key using two different PUFs given the huge size of pairs. However, if a third-party source is considered, then it might be possible to share the same key across different devices.

However, this method is not feasible due to the extensive time required to discover the challenges corresponding to the key in both the PUFs. This problem can be solved by the latest configurable PUFs. These can be easily managed to create the same key on different nodes, and it can generate a large enough number of shared keys for IoT applications.

2.5 | Cross-factor verification and key agreement protocol

The most widely used verification method is a password, but it has its cons. They are easy to guess and usually, people keep the same password for multiple things one more problem which arises is that devices are required to store some details to verify the password. Hence it is prone to physical theft as well as phishing attacks making a password, not a strong authentication protocol.

In the meantime, usage of password only has failed to deliver the required security, it was then suggested to practice a two-factor authentication scheme. A combination of smartcards and passwords was suggested to improve efficiency and anonymity but it was prone to physical theft of smartcards and guessing the password. The same concept was adopted for vehicles by Mohit et al.,[44] but it was later found in Reference 12 that it was unable to guarantee anonymity and robust security.

A three-factor authentication mechanism can provide more robustness to the security system. The device can be accessed only if all the three factors such as password, biometrics, and a tool which is usually a smartcard have been verified. A two-factor authentication scheme can be upgraded to three-factor authentication using a fuzzy extractor. Instead of using smartcards, mobile devices or fuzzy vaults can also be used as a factor along with biometrics and password. An interesting algorithm of bio hashing was used for the security and privacy of telemedicine servers. An AKE-based architecture uses tickets to access the device. To get the ticket the user must verify his identity. To ensure biometrics privacy it was proposed that encrypting the biometrics before transmitting can also use AKE to achieve three-factor authentication. But none of the papers focused on including PUF as a factor for three-factor authentication.

2.6 | Importance of the proposed work

Table 1 compares different attributes of security amongst the relevant protocols.[44–46] It is evident from the table that our proposed CROPUF provides all the required attributes of the security, on the other hand the other related protocols[44–46] fail to achieve physical attacks and desynchronization resilience. The table approves of our proposed CROPUF of delivering few supplementary security attributes such as resiliency towards desynchronization and physical attacks as compared to other protocols. After observing the results of this table, we can summarize that the proposed CROPUF has accomplished extra features of security and has outdone all the other relevant protocols in regard with the features such as computation overhead. However, the other feature, that is, the overhead in communication of our proposed protocol is somewhat larger than[44]; yet, the trade-off amongst extra features of security and the communication overhead recovers this concern.

The Table 2 presents a comparative analysis of the introduced work with the other connected protocols. The security attributes are listed on the basis of which the proposed protocol is measured with the existing state-of-the-art works. The attributes enlisted in the table are discussed as follows:

TABLE 1 Comparison based on security features.

Attribute	Ma et al.[45]	Li et al.[46]	Mohit et al.[44]	Proposed
Physical attack resilience	×	×	×	✓
Vehicle anonymity	✓	✓	✓	✓
Vehicle traceability	×	✓	✓	✓
Mutual authentication	✓	✓	✓	✓
Desynchronization attack resilience	×	×	×	✓
Vehicle impersonation attack resilience	✓	NA	✓	✓
Resilience to infrastructure impersonating attack	NA	NA	✓	✓
Resilience to trusted authority impersonating attack	NA	NA	✓	✓
Perfect forward secrecy	×	×	×	✓

TABLE 2 State-of-the-art comparative analysis with the proposed protocol.

Attributes	47,48	49	50,51	52,53	54–56	57–60	Proposed
Decentralized	✓	✓	✓	✓	✓	×	✓
Guaranteed security	✓	×	×	×	✓	✓	✓
Physical protection	×	×	×	×	×	×	✓
Vehicle Registration	✓	✓	✓	×	×	×	✓
Unique vehicle ID	×	×	✓	×	✓	×	✓
Privacy of vehicle	×	✓	✓	×	✓	✓	✓
Certificate use	×	×	✓	×	×	×	✓
Scalable	×	×	×	×	×	×	✓
Low-latency	✓	×	×	×	×	×	✓

1. *Decentralized*: The degree of decentralization is measured by the span of the geographical distribution of aggregators.
2. *Guaranteed security*: The protocol analyses 51% attacks and justifies the security attributes.
3. *Physical protection*: PUFs are utilized for protecting the vehicles of any physical attacks.
4. *Vehicle registration*: The mutual authentication attribute is provided by the proposed protocol.
5. *Unique vehicle ID*: The vehicle is provided with its own unique PUF.
6. *Privacy of vehicle*: The vehicle privacy is ensured using secure key sharing mechanism.
7. *Certificate use*: Certificates are used to conserve privacy and reduce overhead of communication.
8. *Scalable*: Dynamic consensus is utilized to show throughput scaling.
9. *Low-latency*: Use of aggregators as a fog layer communication is utilized which will reduce load and improve the efficiency of communication between vehicles and the data center.

Therefore, we can conclude that the proposed method is centered around enhancing user authentication and security through a three-factor authentication protocol CROPUF amalgamating biometrics, and passwords. This combination aims to provide a higher level of security against both eavesdropping attacks and physical tampering.

- Three-factor authentication: Since the proposed method combines three distinct factors for user authentication: CROPUF, biometrics (such as fingerprint or facial recognition), and passwords, it is significantly harder for attackers to compromise the system. Each factor adds an additional layer of security, making it more difficult for unauthorized individuals to gain access.
- In addition, fuzzy commitment is a cryptographic technique that accounts for variations or noise in the data. In this regard, it is used to handle any kind of inconsistencies or inaccuracies in the CROPUF responses and biometric measurements. This helps to ensure that authentication is still successful even if the input data is not exactly the same every time.
- Further, the proposed method has also proven to be cost-effective taking both less effort and resources in creating shared keys, which are then generated and used as session keys for encrypting and decrypting the confidential information that would be shared eventually.
- To introduce fog layer systems, Aggregators (intermediary systems) are utilized that act as charging points and enhance communication. These systems serve as a fog layer, which is a concept in edge computing. Fog computing involves distributing computation, communication, and storage closer to the edge of the network, reducing latency and improving performance. These aggregators can also contribute to the overall security and efficiency of the communication between data centers and vehicles.

3 | PRELIMINARY

3.1 | PUF

A physical unclonable function is a physical object that provides a physically defined output (response) for a given input and conditions (challenge). This output or response for a given input serves as a unique identity for the recognition of the IoT device. PUFs are most often based on unique physical variations which occur naturally during semiconductor manufacturing. The challenge and response pair are both bit strings and essentially

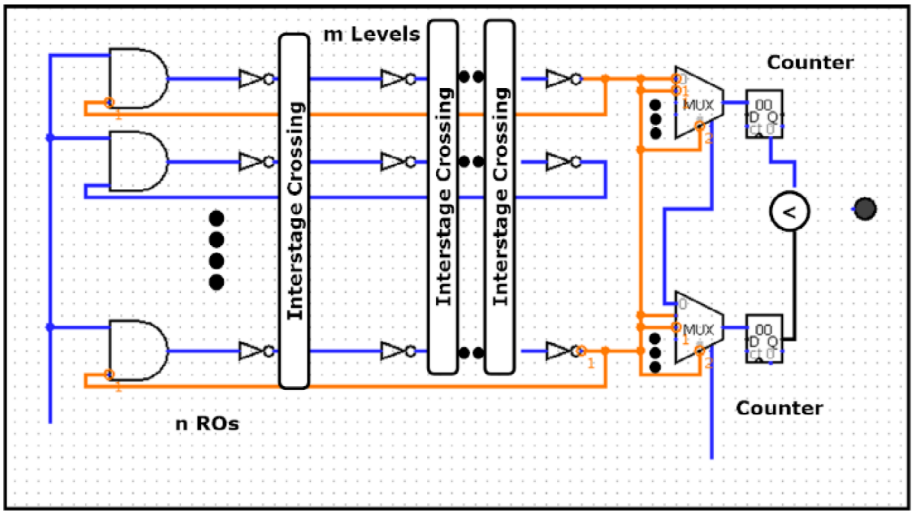

FIGURE 3 Crossover RO PUF structure.

it works as a one-way function $R = PUF(C)$. It is impossible to clone a device and hence the CRP stays unique. An attempt to alter physically abuse the device will in turn alter the response failing the device.

3.2 | Configurable PUF

Crossover RO PUF was presented to enhance the reliability of RO PUF since it has a small number of CRPs. The two inverters are selected by 2×1 multiplexer at every stage. Hypothetically, an RO-based PUF has n number of ROs and m number of level of inverters then every RO is made up of m inverters that operate at a certain rate. The value of m is set to be not less than 2, else RO would fluctuate too quickly for the counter to count, m also must be odd. After interstage crossover, the yields of the previous level inverter are provided as the inputs to the following level of the inverter for m levels of inverters. The path followed by the signal is decided without any further computation by the cross-stage cell. For varying the delay loop design which takes selection inputs, the $m - 1$ interstage crossover cells are there. Figure 3 shows the configuration selection $S = (S_1, S_2, I, S_i, I, S_{m-1})$, where S_i has $\log_2 n$ bits and in the ith stage S_i decides the connection to the upcoming inverter. Finally, the challenge and S are merged to form the complete challenge which is then sent as an input to CRO PUF for response generation. S_{m-1} is committed to maintaining closed loops.

The delay loops can be configured in $\left(A_n^n\right)^{m-2}$ number of ways. The value of m does not have a direct relationship with the value of n. The number of potential challenges is determined by the value of n, while the frequency level of the Ros is determined by the value of m. The value of m decides the number of inverters, if the value is large, it implies the value of frequency would be low. The value of frequencies should not be excessively large or small in real-world applications. Higher frequency would necessitate the use of a high precision counter, on the other hand, low-frequency responses take a huge amount of time to generate, which harms the hardware as the overhead would be high. Hence to meet the criteria the value of m is set between 5 and 7.

3.3 | BCH algorithm

The BCH codes, also known as Bose–Chaudhuri–Hocquenghem codes, are a type of cyclic error-correcting code that is built using polynomials over a finite field in coding theory. One of the vital features of BCH codes is that the amount of symbol mistakes that the code can fix can be controlled exactly while during constructing the code. Binary BCH codes that can fix numerous bit faults can be created. The BCH codes are simpler in decoding, and they are attained using a numerical method called syndrome decoding. It simplifies the design of the decoder for particularly these codes, hence letting a compact use, low-power electrical devices. BCH encoding combined with fuzzy extraction makes sure the noise in inputs is taken care of and the user can get access to sensitive data without having to worry about the physical conditions which might impact the PUF or biometrics data.

BCH encoding is merely the process of finding some polynomial that has the generator as a factor. Therefore, there are many ways to encode a given binary string. Primarily the two methods which are used are discussed here (Algorithm 1).

Algorithm 1. BCH encoding

Input: A finite field GF(q) where q is a prime power, consider q=2, and hence all the polynomials to be represented as GF(2). Generator polynomial $g(x) = x^{10} + x^9 + x^8 + x^5 + x^3 + x^2 + 1$, in the form of an array of size 32 i.e., $genPoly = \{1,0,1,1,0,1,0,0,1,1,1,0,0 \ldots\}$, input binary string $messageString = \{10110111011111011111101\}$

Output: Encoded code (binary string)

Procedure 1: Non-systematic encoding: The message as a factor:

1. $message \leftarrow array\ of\ size\ 32$
2. $messageLength \leftarrow size\ of\ string\ messageString$
3. **For** $i = 0,1,2 \ldots\ to\ messageLength$ **do**
4. **If** $(messageString[i] = 1)$
5. $message[32 - i - 1] \leftarrow 1$
6. **End if**
7. **End for**
8. $prod \leftarrow array\ of\ size\ 32$
9. **For** $i = 0,1, \ldots\ to\ 32$ **do**
10. **For** $j = 0,1,2 \ldots to\ messageLength$
11. $prod[i + j] \leftarrow prod[i + j] + message[j] * genPoly[i]$
12. **End for**
13. **End for**
14. $encodedString \leftarrow string\ of\ size\ 32$
15. **For** $i = 0,1,2, \ldots to\ 32$ **do**
16. **If** $(prod[i] \geq 1)$
17. $encodedString[32 - i - 1] \leftarrow prod[i]$
18. **End if**
19. **Return**: $endcodedString$

Procedure 2: Systematic encoding: The message as a prefix

1. $message \leftarrow array\ of\ size\ 32$
2. $messageLength \leftarrow size\ of\ string\ messageString$
3. **For** $i = 0,1,2 \ldots\ to\ messageLength$ **do**
4. **If** $(messageString[i] = 1)$
5. $message[32 - i - 1] \leftarrow 1$
6. **End if**
7. **End for**
8. $genPolyLength \leftarrow 0$
9. **For** $i = 0,1,2 \ldots to\ 32$ **do**
10. **If** $(genPoly[i] = 1)$
11. $genPolyLength \leftarrow i + 1$
12. **End if**
13. **End for**
14. $tempDiv \leftarrow Array\ of\ size\ 33$
15. $quotient \leftarrow Array\ of\ size\ 32$
16. $quotientLength \leftarrow messageLength - genPolyLength$
17. **While** $(genPolyLength < messageLength)$ **do**
18. **For** $i = 0,1,2 \ldots to\ genPolyLength$ **do**
19. $tempDiv[i + 32 - genPolyLength] \leftarrow genPoly[i]$
20. **End for**
21. $quotient[messageLength - genPolyLength] \leftarrow message[messageLength]/genPoly[genPolyLength]$
22. **For** $i = 0,1,2 \ldots to\ quotientLength + 1$ **do**
23. $tempDiv[i] \leftarrow tempDiv[i] * quotient[messageLength - genPolyLength]$
24. **End for**
25. **For** $i = 0,1,2 \ldots to\ messageLength + 1$ **do**
26. $message[i] \leftarrow message[i] - tempDiv[i]$
27. **End for**
28. $messagelength \leftarrow messagelength - 1$
29. **End while**
30. $prod \leftarrow array\ of\ size\ 32$
31. **For** $i = 0,1, \ldots\ to\ quotientlength$ **do**
32. **For** $j = 0,1,2 \ldots to\ genPolyLength$
33. $prod[i + j] \leftarrow prod[i + j] + quotient[i] * genPoly[j]$
34. **End for**
35. **End for**
36. $encodedString \leftarrow string\ of\ size\ 32$
37. **For** $i = 0,1,2, \ldots to\ 32$ **do**
38. **If** $(prod[i] \geq 1)$
39. $encodedString[32 - i - 1] \leftarrow prod[i]$
40. **End If**
41. **End For**
42. **Return**: $endcodedString$

In a similar fashion, there are numerous ways for decoding an encoded BCH code however, most of them follow the following algorithm with some specific decisions in every step (Algorithm 2).

Algorithm 2. Decoding BCH

Input: The received vector R is the sum of the correct codeword C and an unknown error vector E. Generator polynomial $g(x)$ with roots α^j.
$array\ syndromes[] \leftarrow syndromeCalculate\ (R, g(x), \alpha^j)$
Output: Corrected Code

1. **If** $syndromes.empty()$
2. **Return**: **R**
3. **End if**
4. **Else**
5. $int\ numberOfSynromes \leftarrow syndromes.length()$
6. **If**$(numberOfSynromes = 1)\{$
7. $errorPosition \leftarrow errorLocator(syndromes, \alpha^j)$
8. **End If**
9. **Else**
10. $var\ locatorPolynomial \leftarrow calculateLocatorPolynomial(syndromes, \alpha^j)$
11. $array\ errorPositions[] \leftarrow errorLocator(syndromes, \alpha^j, locatorPolynomial)$
12. **End Else**
13. $array\ errorValues[] \leftarrow calculateErrorValues(errorPositions, syndromes, \alpha^j)$
14. $correctedCode \leftarrow ErrorCorrector(errorValues, errorLocator, R)$
15. **End Else**
16. **Returns**: CorrectedCode

3.4 | Fuzzy extractor

Fuzzy extractors are a security approach that allows the data to be used as inputs to conventional cryptography algorithms. The term "fuzzy" refers to the fact that the fixed values will be similar to but not identical to the original key while maintaining the appropriate level of security. Fuzzy extractors are extremely useful for PUF as the response to a challenge can change slightly due to environmental conditions. Hence, they can help with the issue of noise. Figure 4 shows the overall procedure followed in fuzzy extraction. The process is divided in two phases:

Generation (Gen): In the generation phase, our input data serves as a key *w* which is usually a string of bits for extracting a random and unique string *K* and a helper string *P*. Moreover, generation is a probabilistic algorithm.

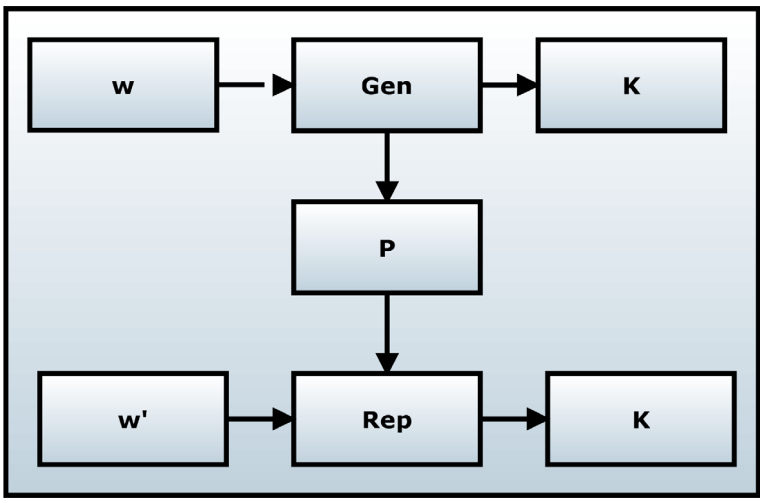

FIGURE 4 Fuzzy extraction.

Reproduction (Rep): Given the helper string P and slightly modified input (w', given w and w', do not differ by a given amount, usually distance refers to the hamming distance) we can recover K. Reproduction is a deterministic algorithm.

3.5 | Feature fusion based on fuzzy commitment

Usually, the fuzzy extractor utilizes the response coming from PUF I or input such as Biometrics (B) nonetheless, in this article, we combine both of them to achieve a two-factor authentication. The procedure is as follows:

1. Enrolment (B, R) – {K, P} K and P are unique key and helper data, respectively.

   - We select a random string N of bits and encode it with the BCH algorithm.
   - We encrypt the corrected N using SHA-256 or any suitable encryption technique.
   - We fuse the input features, that is, B and R (example XOR) $w = B \bigoplus R$.
   - The output is finally calculated using encoded N and w, $P = w \bigoplus Encoded(N)$.

   Figure 5 highlights the procedure to enroll a user's biometrics and his unique response from PUF.

2. Reproduction (P, R', B') – {K'} R' and B' are slightly modified inputs for the reproducing of key (K'). The detailed procedure is depicted in Figure 6. The steps involved in the procedure are as follows:

   - We compute $N' = Decode\left(P \bigoplus B' \bigoplus R'\right)$.
   - We encrypt N' to reproduce K'.

4 | PROPOSED SCHEME

By integrating all three factors such as PUF, biometrics, and password, the proposed system achieves a three-factor security mechanism, all of which execution is done mainly executed on the user side. The noise from the response of the PUF is administered by a fuzzy extractor on the vehicle sensor side. The fuzzy commitment is used on the user side to unravel the issue of noise in the Biometric and response of PUF. Furthermore, the proposed approach uses feature fusion to integrate biometrics and devices PUF response features to deliver a stronger safety for IoT devices.

The four participants in the scheme are:

- User (U). The user receives assistance for managing vehicles and routing them using the real-time sensory data, as well as data from other vehicles available in data centers.
- Data center (DC). The data center oversees managing and storing the data gathered by different vehicle sensors in a particular region.

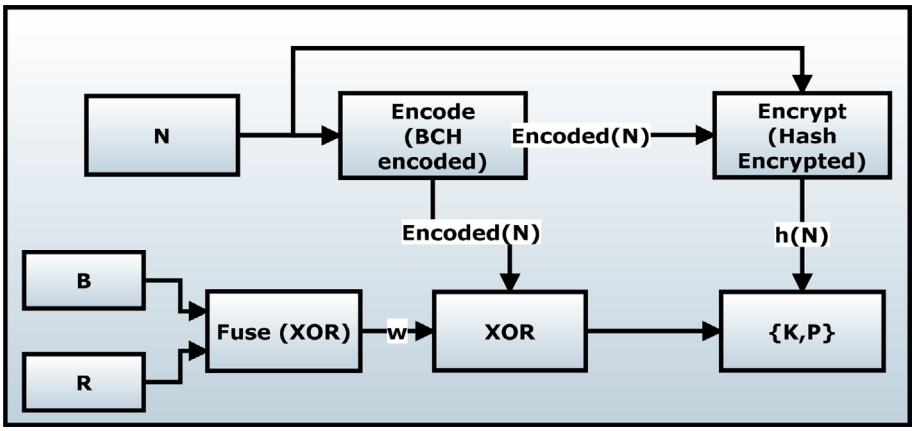

FIGURE 5 Enrolment phase.

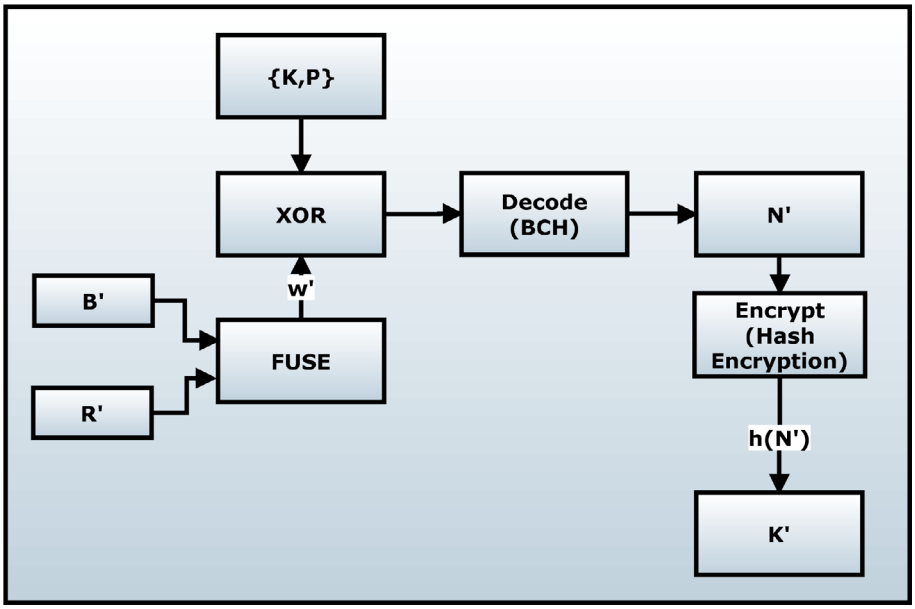

FIGURE 6 Reproduction phase.

- Vehicle sensor (*V S*). It is installed in the car itself to gather real-time data of various things like traffic, physical environment, and so forth.
- Aggregator (*Agg*).

The important notations are mentioned in Table 3. The suggested scheme has six main steps:

1. System setup
2. Registration
3. Verifying aggregator to data center
4. Login and authentication
5. Password update
6. Biometric update

TABLE 3 Symbols used.

Symbols	Description
U_{PK}, D_{PK}, V_{PK}	The identification of *U*, *DC* and the vehicle, respectively
PWD, BIO_m	The password and biometrics of *U*
UPUF, *VPUF*	The PUF of *U* and *V S* respectively
(*UC*, *UR*) (*VC*, *VR*)	The PUF pair of challenges and its respective responses of *U* and *V S*
D, *d*	The public/private key pair of *DC*
VPU	The public key of *V S*
Encode(·), *D*ecode(·)	The methods of encoder and decoder in BCH
Gen(·), *Rep*(·)	Probability generator and deterministic producer function.
HASH(·)	The one-way hash function (SHA-256) of cryptography
Ti ($i \in [1, 4]$)	The time logs are utilized in the login and verification phase
||	Concatenate
$\oplus$	Bitwise XOR

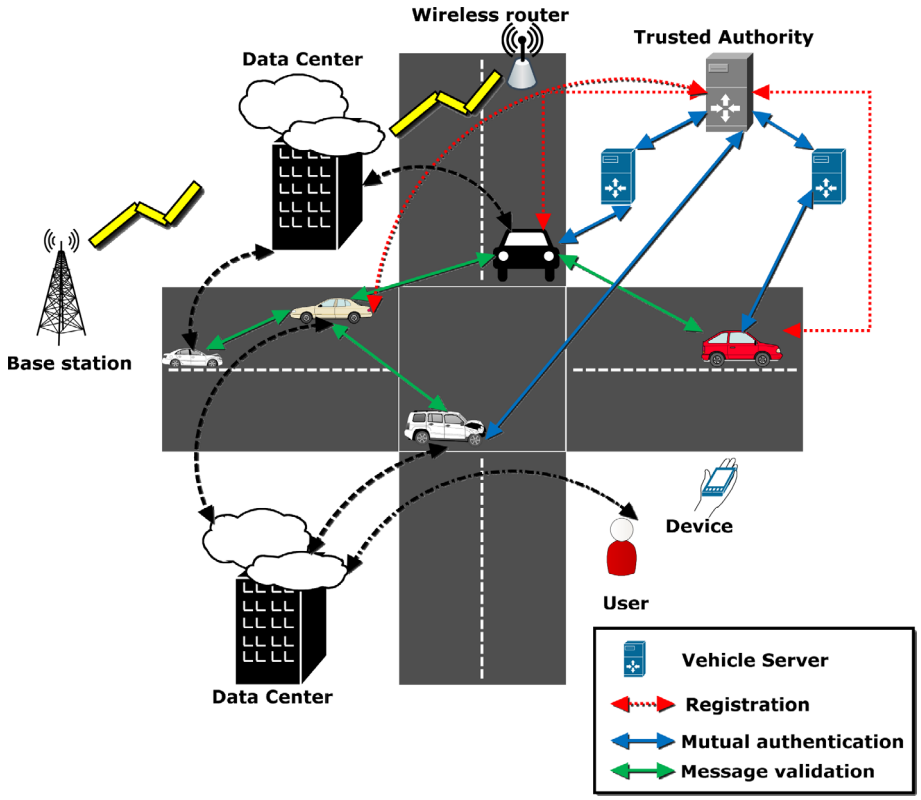

FIGURE 7 V2V communication scenario.

Figure 7 illustrates the basic system model of a typical V2V communication system comprising of various entities that form the system. The system model shows three phases namely: the registration phase, the communication phase and the mutual authentication phase. In the registration phase, vehicles get themselves registered through secure channels. During the communication phase, authentication of all the entities such as TA, vehicles, RSUs takes place. The session keys are generated for them to communicate with each other. For the intra-region communication between the vehicles, the session key is shared via a local RSU while in case of inter-region communication between the vehicle's communication starts after RSUs exchange session keys amongst themselves via TA. We consider TAs and RSUs as reliable nodes. In addition, we deliberate some assumptions such as:

- TA is trusted entity.
- V2V communication allows only registered vehicles.
- Passwords are not shared by legitimate vehicle users with bogus users.

In this scenario of V2V communication, the vehicle uploads its driving data like speed of the vehicle, conditions of the road, status of the engine, and so forth, which is collected by the Vehicle Sensor (VS) and sent to the data center (DC). The statistical data is accessed from DC by the users such as drivers and transportation authorities and they can also access time-sensitive data from VS. This helps in effective route planning and managing vehicles.

4.1 | PUF-based key-sharing

4.1.1 | Principle of shared-key generation

In multiparty communication, it is ideal to have a shared key amongst different devices. With traditional PUFs, this is not possible as they create chip unique keys, but CRO PUF is capable of generating the same shared key making them ideal for such types of communication. With the right setups and challenges, multiple components can generate the identical replies as the shared key. The number of selection signals and challenges rises exponentially as *n* and m are increased. Furthermore, the variety of interstage crossover structure options allows for considerable flexibility in

authentication.

$$delay_{RO}\begin{bmatrix} d_{11} & \cdots & d_{1m} \\ \vdots & \ddots & \vdots \\ d_{n1} & \cdots & d_{nm} \end{bmatrix}.$$

The delay vector model having n vectors and every vector is having m inverters is depicted above.

The delay vector of each line $D_{RO} = \{D_1, D_2, \ldots, D_i, \ldots, D_n\}$, where $D_i = \sum_{j=1}^{m} d_{ij}$.

The selection signal directs the path for connection between the jth and $(j+1)$th column inverters, that is,

$$\left\{d'_{1j}, d'_{2j}, \ldots, d'_{nj}\right\} = f\left(d_{1j}, d_{2j}, \ldots, d_{nj}\right).$$

The challenge C_{RO} is used to select the rows, that is,

$$\left\{D'_1, D'_2\right\} = g(D_1, D_2, \ldots, D_n).$$

We utilize the control signal along with a function f for the modification of our delay vector. Another function g of our challenge is used for comparison by selecting various delay vectors.

Since the functions f and g are independent of each other, we can utilize them to get the same response from two non-identical PUFs. Hence, we can generate a shared key that would be common to the two devices without actually having to share the devices information. We now present an example to illustrate how we can get the key, we take two CROPUFs having four-layer inverters, and with the following delay matrices.

$$A = \begin{matrix} 3 & 6 & 8 & 5 \\ 9 & 7 & 4 & 5 \\ 5 & 4 & 6 & 5 \\ 2 & 5 & 6 & 3 \end{matrix} \quad B = \begin{matrix} 2 & 4 & 6 & 5 \\ 5 & 1 & 3 & 2 \\ 8 & 6 & 5 & 7 \\ 3 & 6 & 4 & 5 \end{matrix}.$$

Consider the following challenges:

$$C_A = \{\{1,2\},\{2,3\},\{3,4\}\},$$

$$C_B = \{\{1,3\},\{3,4\},\{4,2\}\}.$$

Hence for the above case the response in both of these devices would be {0, 1, 1}

$$C_A = \{\{4,2\},\{2,3\},\{3,1\}\},$$

$$C_B = \{\{1,2\},\{2,4\},\{4,3\}\}.$$

Now for this case to get the common key we will let the signal for A be same and for B we would modify $S2, S4$.

$$A = \begin{matrix} 3 & 6 & 8 & 5 \\ 9 & 7 & 4 & 5 \\ 5 & 4 & 6 & 5 \\ 2 & 5 & 6 & 3 \end{matrix} \quad B = \begin{matrix} 2 & 4 & 6 & 5 \\ 5 & 6 & 3 & 7 \\ 8 & 1 & 5 & 5 \\ 3 & 6 & 4 & 2 \end{matrix}.$$

After changing the values, the delays are as follows

$$D_A = \{22, 25, 20, 16\},$$

$$D_B = \{17, 21, 19, 15\}.$$

For the above-mentioned changes, the response for both devices would be {0, 1, 0}. It is worth noting that overall delay is also dependent on the structure of our crossover. The field programmable gate array (FPGA) tool helps to minimize delay variance in two routes however it essentially cannot get exactly the same architecture for the equal delay. The delay is dependent on the length of the route in an RO PUF. Therefore, it is possible for our PUF circuit to be altered or modified during the process of production, if the multiplexers and inverters used cannot induce enough delay to offset the delays caused by different routes. But earlier it was proved that by setting SRAM to desirable values the PUF can function properly without any issues. It was also reported that PUF had a high value of uniqueness, that is, the crossover's delay has almost no effect. But it is interesting to note that the efficiency and robustness can be increased even further by including the model for the structure in the PUF.

4.1.2 | Modelling of delay matrix

To obtain the delay matrix from a given PUF we make use of Machine learning. We try to obtain a delay matrix that is very close to the actual delay matrix however, it is difficult to recover the actual matrix.

Process of machine learning algorithm

The process is fairly simple and contains just two steps: we list all the delay paths to get the values of counters in the actual delay matrix, then we predict the delay matrix using a model that we build. We use different challenges to choose two different delay paths then compare them. Figure 8 shows what a delay path would look like in an actual PUF but it is not possible to predict the real values just by comparing hence we store the Count: values of delay parameters. The value of count can be calculated as follows:

$$Count = \frac{1}{C \cdot D},$$

where, D is the delay matrix.

To model the PUF and obtain the counter values the PUF has an interface which is made using fuses, after the model is extracted by the designer, they destroy the fuses to ensure that no adversary can gain access to actual model. This is done before the distribution of chips to ensure security.

4.1.3 | Shared-key generation

Consistent generation of response for shared-key

To obtain the mutual key, our CRO PUF must provide responses that are stable and robust to different challenges. Since we already know that we can get the delay matrix with using machine learning with high accuracy, we can easily get difference in delay of any two paths. Subsequently, we sort these absolute values differences in non-increasing order for determining the threshold value T. T is defined as the value for which the response is stable at variety of temperatures if the difference in delays of two paths is greater than T. The Algorithm 3 shows how to select an appropriate threshold value.

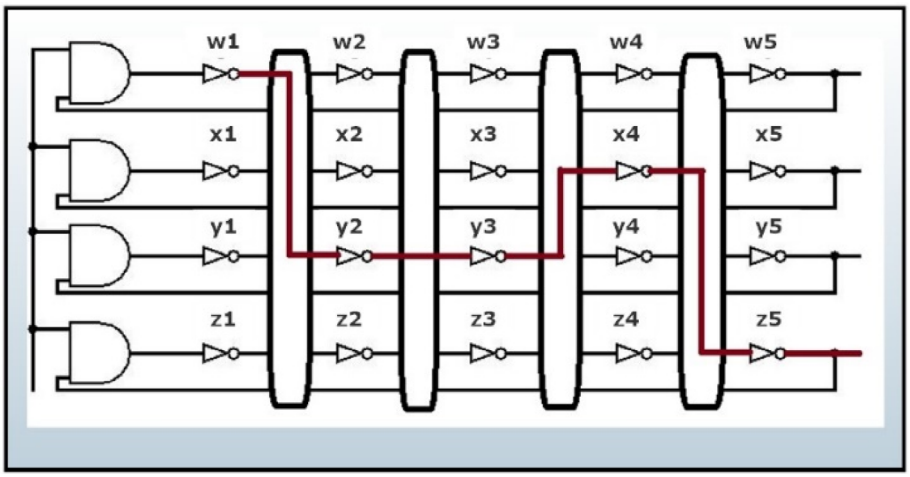

FIGURE 8 Example of a delay path.

Algorithm 3. Get the threshold

Input: The matrix of Delays A
Output: The value of Threshold T

1. **Reset**: D: conceivable path delays set in A
2. **for** $d1 \in D, d2 \in D$ and Path ($d1$) ≠ Path ($d2$) do
3. $B \leftarrow (|d1 - d2|$, Config ($d1, d2$)) $\cup B$
4. //Config (d1, d2): the configuration challenge of $d1$ and $d2$
5. **end for**
6. Sort (B) // sort the matrix in descending order
7. N = Sizeof (B)
8. **For** $int\ i = 1, 2, \ldots, N$ ***and*** $(Xi, Yi) \in B$ **do**
9. // Xi: the difference in delays
10. // Yi: the configuration challenges
11. **if** Yi produces a response that is not sable at various temperatures **then**
12. $T = X$//X: a value considerably larger than Xi
13. **break**
14. **end if**
15. **end for**
16. **return** T

Algorithm 4. Generation of challenge

Input: Delay matrix D, Threshold T, Mutual key PK
Output: Challenge C

1. **Initialize**: $D \leftarrow all\ possible\ delay\ paths$
2. $n \leftarrow sizeof(PK)$
3. $m \leftarrow sizeof(D)$
4. **for** $i = 1, 2, \ldots, N$ ***do***
5. $r1 \leftarrow rand(0, m), r2 \leftarrow rand(0, m),$
6. $p1, p2 \leftarrow D[r1], D[r2]$
7. **while** $path(p1)! = path(p2)$ **do**
8. **if** $PK[i] == 1$ && $p1 - p2 > T$ **then**
9. $C[i] = Config(p1, p2)$
10. **break**
11. **Else if** $PK[i] == 0$ && $p1 - p2 < -T$ **then**
12. $C[i] = Config(p1, p2)$
13. **break**
14. **End if**
15. **End while**
16. **End for**
17. **Return** C

Given that our aim is not only to make stable responses, but also to utilize the generated responses as shared keys for IoT devices, for which we introduce an algorithm designed to generate challenges for the PUF. Algorithm 4 describes the process of challenge generation. The unreliability in responses of PUF can also be caused due to various external factors over which we have no control such as instability in temperature, dielectric dependent on temperature, and so forth. People have started to realize that external factors can greatly impact the performance and hence new scholars are trying to mitigate this issue by proposing various solutions. Along with that ageing resistance is an important key issue in the current PUF related operations.

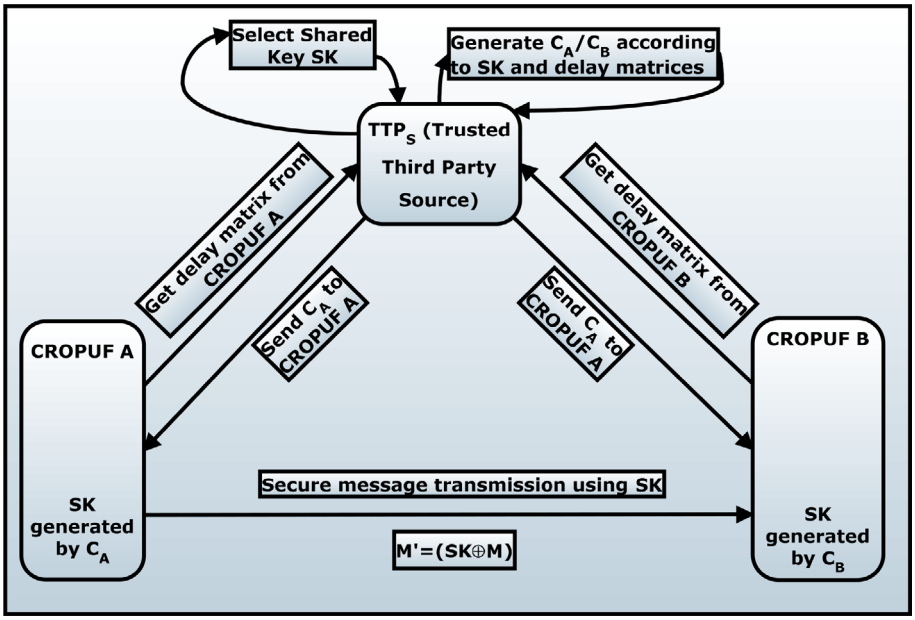

FIGURE 9 Key-sharing on the basis of PUF and protocol for confidential transmission of information.

Challenge generation for shared-key

The next section covers the exact procedure for getting the shared key in two or more devices using a common source, the source must be trusted one, that will do the entire computation on the delay matrices and generate challenges as well. Algorithm 4 shows the steps required to generate challenge from delay matrix after selecting an appropriate key. The algorithm selects two distinct paths and checks if the difference in the delays is above the threshold value that was calculated using Algorithm 3, if yes then we get the configuration for the selected paths, else we start the process again by selecting two new paths.

4.1.4 | Key-sharing protocol

We have already discussed that how we can utilize CRO PUF for getting a shared key between two devices to authenticate. We propose a protocol that incorporates two PUF devices and a trusted third-party source to facilitate the selection of the shared key and the generation of challenges for these devices. Let PUF_A and PUF_B be the two devices that need a shared key and TTP be the third-party source. First the delay matrices of the PUF devices are shared with TTP. Then TTP will select a secret key of appropriate length to select as the shared key. After that according to the delay matrices of PUF_A and PUF_B, TTP uses the above-mentioned algorithms to generate the challenges for both devices C_A and C_B. Then these challenges are shared to the respective devices, finally using these challenges the devices can use their PUF responses to use as the shared key. Throughout this entire process, no personal information was shared using any of the communication challenges, it can be ensured that there is no risk of intervention or eavesdropping. The only private information was about the delay matrices of PUF devices, but they were already stored in our TTP. After receiving the challenges and generating the shared key the devices can encrypt the messages that they wish to share with each other. Figure 9 depicts the process of generating the key and sharing the appropriate challenges with required PUF devices to get responses and use it as the shared key.

4.2 | Registration phase

4.2.1 | User registration

For becoming a valid and tested user and also for becoming a part of the IoV network, a user must first register himself and the vehicle sensor on the *DC*. The steps for the same are shown in Figure 10 and also listed below:

Step 1: *U* inputs his identity (private key) U_{PK} to the device, which in turn sends the same to the data center along with a registration request.

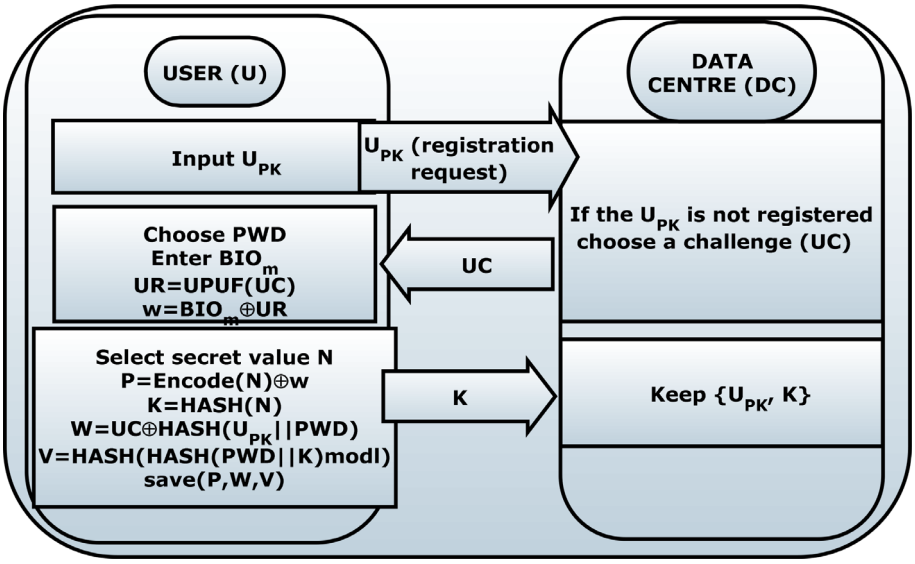

FIGURE 10 User registration phase.

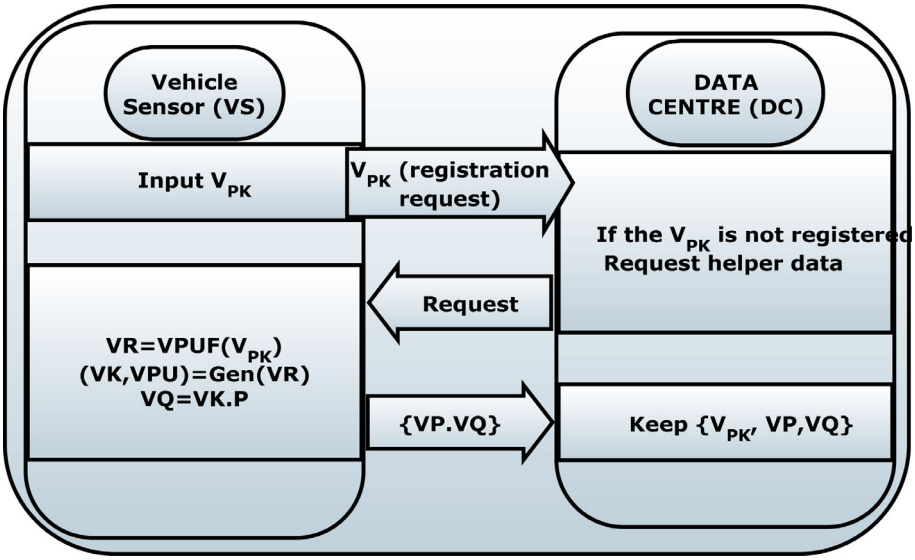

FIGURE 11 Vehicle registration phase.

Step 2: On receiving U_{PK}, *DC* looks through the records of already existing users and checks if the user has been already registered or not if the user is not registered the *DC* generates *UC* and transmits it to *U*. Else, it alerts the user he is already registered.

Step 3: Once the user receives *UC* he selects a *PWD* and imprints his BIO_m onto the device. The user's device then computes $UR = UPUF(UC)$ and utilizes fuzzy commitment fusion technology to induce the biometrics of the user with the device's response of the challenge $w = BIO_m \bigoplus UR$. Furthermore, the device selects a secret value *N* and calculates $= Hash(N)$, $W = UC \bigoplus Hash(U_{PK} \bigoplus PWD)$, and $V = Hash(Hash(PWD \bigoplus K)mod\ l)$. The gadget then saves *P*, *W*, and *V* in its database. Finally, the gadget dispatches *K* to *DC* along with the delay matrix of the PUF.

Step 4: *DC* stores the item $\{U_{PK}, K\}$ into its database for later authentication.

4.2.2 | Vehicle sensor registration

In the vehicle sensor registration phase, the vehicle identification V_{PK} serves as the PUF challenge, and there is no requirement of data retention in vehicle sensor because of the involvement of PUF as illustrated in Figure 11. The following sections provide more information on the exact steps:

Step 1: *VS* sends the registration request by transmitting V_{PK}.

Step 2: On receiving V_{PK}, *DC* looks through the records of already existing vehicle sensors and checks if the *V S* has been already registered or not if the *V S* is not registered the *DC* transmits this information to *V S*. Else, it alerts the user he is already registered.

Step 3: *VS* extracts the PUF output $VR = VPUF(V_{PK})$ by using V_{PK} as the PUF's challenge. *V S* then utilizes the fuzzy extraction to analyze the noise of PUF response and create an assistant data $(VK, VPU) = Gen(VR)$ before computing $VQ = VK.P$ *V S* then transfers $\{VP, VQ\}$ to *DC*.

Step 4: *DC* stores the item $\{V_{PK}, VPU, VQ\}$ into its database for subsequent authentication.

4.2.3 | Aggregator registration

In the registration phase of the aggregator, the aggregator sends CROPUF's public key and delay matrix to the data center. The data center stores this information in its database for generating the shared keys to support further communications.

4.3 | Verification phase

4.3.1 | Aggregator verification/shared key generation

To verify the aggregator from DC. The following steps are involved:

Step 1: Aggregator sends its public key to the data center.

Step 2: On receiving AG_{PK}, the data center verifies whether the key already exists in its database, if it does then *DC* selects a shared key of appropriate length to use for encrypting communication between vehicle and aggregator, after selecting the key it generates appropriate challenge for the same, which it sends over back to the aggregator. If the AG_{PK} received is not present in its database, it rejects the request.

Step 3: After receiving the challenge aggregator generates response using its ROPUF to get the shared key to decrypt any further communication with the data center. Figure 12 shows a diagrammatic representation of the exact steps involved in verification for DC communication.

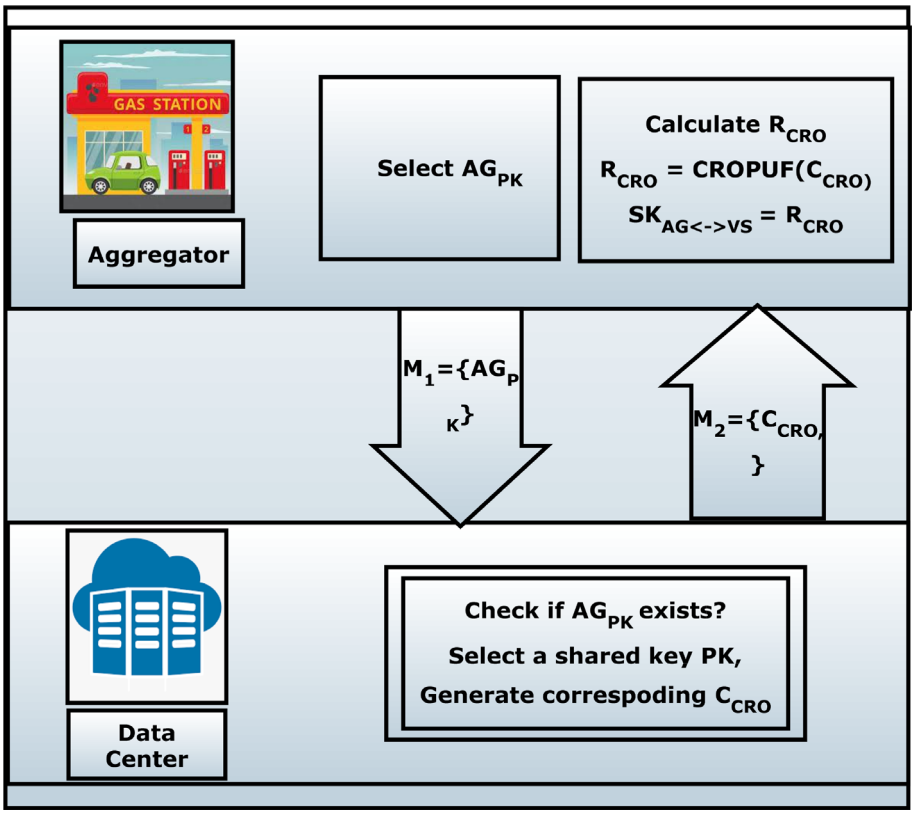

FIGURE 12 Verifying aggregator to DC communication.

4.3.2 | Generating shared key for communication to aggregator

To gain access of the shared key, the sensor of vehicle and user send their respective public keys to the Data Center that generates the challenge equivalent to the same shared key as that used above. After getting the challenge the user and vehicle encrypt all the messages further sent to aggregator using the same key.

4.3.3 | Logging in and verifying *VS* and *U*

For accessing *DC* and *V S*, a user *U* must first log in and obtain the approval of *DC*. After that, he or she must create session keys with *DC* and *V S*, respectively. Figure 13 depicts the exact procedure.

Step 1: *U* initially inputs the device's identification U'_{PK} password PWD'_{new} and biometrics BIO'_m. Then the device extracts the PUF challenge $UC' = W \bigoplus Hash(U'_{PK}||PWD')$ and then calculates the response $UR' = UPUF(UC')$ using the built-in PUF. Following that, the gadget utilizes fuzzy commitment biometric fusion technology to induce the biometrics of the user with that of the component. $w' = (BIO'_m \bigoplus UR')$, $N' = Decoder(P \bigoplus w')$ then it computes $K' = Hash(N')$ to check whether $Hash(Hash(PWD'||K')mod\ I)$ equals *V* stored in local memory for verification.

Step 2: If the value computed beyond equals V, the user component computes its temporary public key $UQ = UN \cdot P$ and the temporary key $B = UN \cdot D = (B_x, B_y)$ shared with DC using a random number *UN*. The device then generates the pseudo-identities $DU_{PK} = U'_{PK} \bigoplus B_y$, $DVPK = V_{PK} \bigoplus B_y$, and a verification message $V_1 = Hash(U'_{PK}||DC_{PK}||V_{PK}\ ||K'\ ||\ T_1||B_x)$, where V_{PK} is the identity of the car that *U* wishes to access and T_1 is the exact time at which the computations are made. Then the device transmits $M_1 = \langle UQ, DU_{PK}, DV_{PK}, V_1, T_1\rangle$ to *DC*.

Step 3: DC verifies the timeliness of the current communication after getting the message from *U*'s device. Otherwise, the temporary key $B' = d \cdot UQ = (B'_x, B'_y)$ is computed, and the user and vehicle identity are restored $U'_{PK} = DU_{PK} \bigoplus B'_x$, $V'_{PK} = DV_{PK} \bigoplus B'_y$. *DC* then obtains *K* by scanning the database for U'_{PK}. In case of its absence in the database, this implies *U* is an unauthorized operator, and *DC* will dismiss the communication immediately. Or else, *DC* computes $Hash(U'_{PK}||DC_{PK}||V_{PK}\ ||K'\ ||\ T_1||B_x)$. If the computed value is the same as the one saved, DC looks up *VP* and *VQ* in the database using V'_{PK}, and then calculates $J = d \cdot VQ = (J_x, J_y)$ After that, *DC* calculates *U*'s pseudo-identity $DU'_{PK} = U'_{PK} \bigoplus Hash\left(V'_{PK}||DC_{PK}||J_x\middle\|T_2\right)$ and the key material $DK = Hash(U'_{PK}||K||T_2) \bigoplus Hash(J_x||T_2)$, which helps *V S* and *U* compute the key of the session and the material for session key $DCSKs = Hash(DC_{PK}||d||T_2) \bigoplus Hash(J_2||T_2)$ between *DC*. Finally, DC calculates $V_2 = Hash(V'_{PK}||DC_{PK}\ ||U'_{PK}\ ||\ J_x\ ||J_y||\ T_2)$, $M2 = \langle DU'_{PK}, DK, DCSK, V_2, T_2\rangle$ and transmits them to the vehicle sensor *V S*.

Step 4: *V S* instantly double-checks the message's timeliness after receiving it. *V S* utilizes *V ID*′ k to derive $VR' = VPUFk\ (V'_{PK})$ if it is fulfilled. The temporary key $J' = VK' \cdot D = (J\prime_x, J\prime_y)$ is then calculated using the fuzzy extraction to cope with the PUF response disturbance $(VK', VP') = Gen(VR')$. *V S* then calculates $Hash(V_{PK}||DC_{PK}\ ||U''_{PK}||\ J\prime_x\ ||J\prime_y||\ T_2)$, to see if the computed value is identical to the identity received using $U''_{PK} = U'_{PK} \bigoplus Hash(V'_{PK}||DC_{PK}||J_x||T_2)$. *DC*'s identity is accepted if they are equal. Otherwise, *V S* rejects *DC* and ends the session.

Step 5: *V S* calculates session key:
$SK_{V\longleftrightarrow U} = Hash(Hash\ (DU_{PK} \bigoplus Hash(J'_x||T_2))||\ Hash(U'_{PK}||T_3||VK'))$, $SK_{V\longleftrightarrow S} = Hash(Hash\ (DCSK \bigoplus Hash(J'_y||T_2))||\ Hash(V'_{PK}||T_3||VK'))$, which is shared with *U* and *DC*. *V S* then calculates the pseudo response $DVR = VR' \bigoplus J\prime_x$, the verification value $V = Hash(V'_{PK}||DC_{PK}||J'_y||\ SK_{V\longleftrightarrow U}||D||T_3)$, and $V4 = Hash(U''_{PK}||V'_{PK}||VK||\ SK_{V\longleftrightarrow U}||T_3)$ with V_4 being transmitted to *U* via *DC*. *VS* then transmits the message to *DC*.

Step 6: *DC* double-checks the message's timeliness after getting it from *VS*. Then *DC* calculates $VR'' = DVR \bigoplus J_x$, $VK'' = Rep(VP, VR'')$. The shared session key $SK_{S\longleftrightarrow V} = Hash(Hash\ (DC_{PK}||d||T_2)||Hash(V'_{PK}||T_3||VK''))$ and the value $Hash(DC_{PK}||V'_{PK}||SK_{S\longleftrightarrow V}||T_3)$ are then computed using the determined *V S*'s secret value and *DC*'s secret value *d*. Then, by determining if the computed value equals V_3, *DC* validates the identity of *V S* and the authenticity of the key. If they are equal, *DC* generates the session key $SK_{S\longleftrightarrow U} = Hash(Hash\ (DC_{PK}||d||T_4)||Hash(U'_{PK}||T_4||K))$ shared with *U*, as well as a verification value $V_4 = Hash(DC_{PK}||U'_{PK}||B'||SK_{S\longleftrightarrow U}||\ T4)$. *DC* then calculates the pseudo-secret value $AVK = B\prime_y \bigoplus VK'$ and pseudo-key materials $DVK = Hash(V'_{PK}\ ||T_3||VK') \bigoplus Hash(T_4||K)$, $DCSK = Hash(DC_{PK}||T_4||d)\ h(K||T_4)$, where T_4 is the current timestamp. Finally, *DC* transmits the message $M4 = \langle AVK, DVK, DCSK, V_4, V_5, T_2, T_3, T_4\rangle$ to *U*.

Step 7: Double checking of the timeliness for the message is conducted by *U* initially after receiving the message. If this is the case, *U* calculates the session key $SK_{S\longleftrightarrow U} = Hash\ (DC_{PK} \bigoplus Hash(K'||T_4))\left\|Hash(U'_{PK}||T_4||K')\right.$ shared with *DC* and $Hash(U'_{PK}||DC_{PK}||B_y||\ SK_{U\longleftrightarrow S}||T_4)$ to assess the validity of *DC* and the session key. If the computed value equals *V*5, Ui retrieves *V S* secret value $VK'' = AVK \bigoplus B_y$ and calculates the shared session key $SK_{U\longleftrightarrow V} = Hash(Hash\ (DV_{PK} \bigoplus Hash(K'||T_4))||\ Hash(U'_{PK}||T_4||K'))$ and the verification message $Hash(U'_{PK}||V_{PK}||VK''||\ SK_{U\longleftrightarrow V}||T_3)$, and determines if the computed value is equal to the received V_4 to ensure *V S*'s identity and the session key's validity.

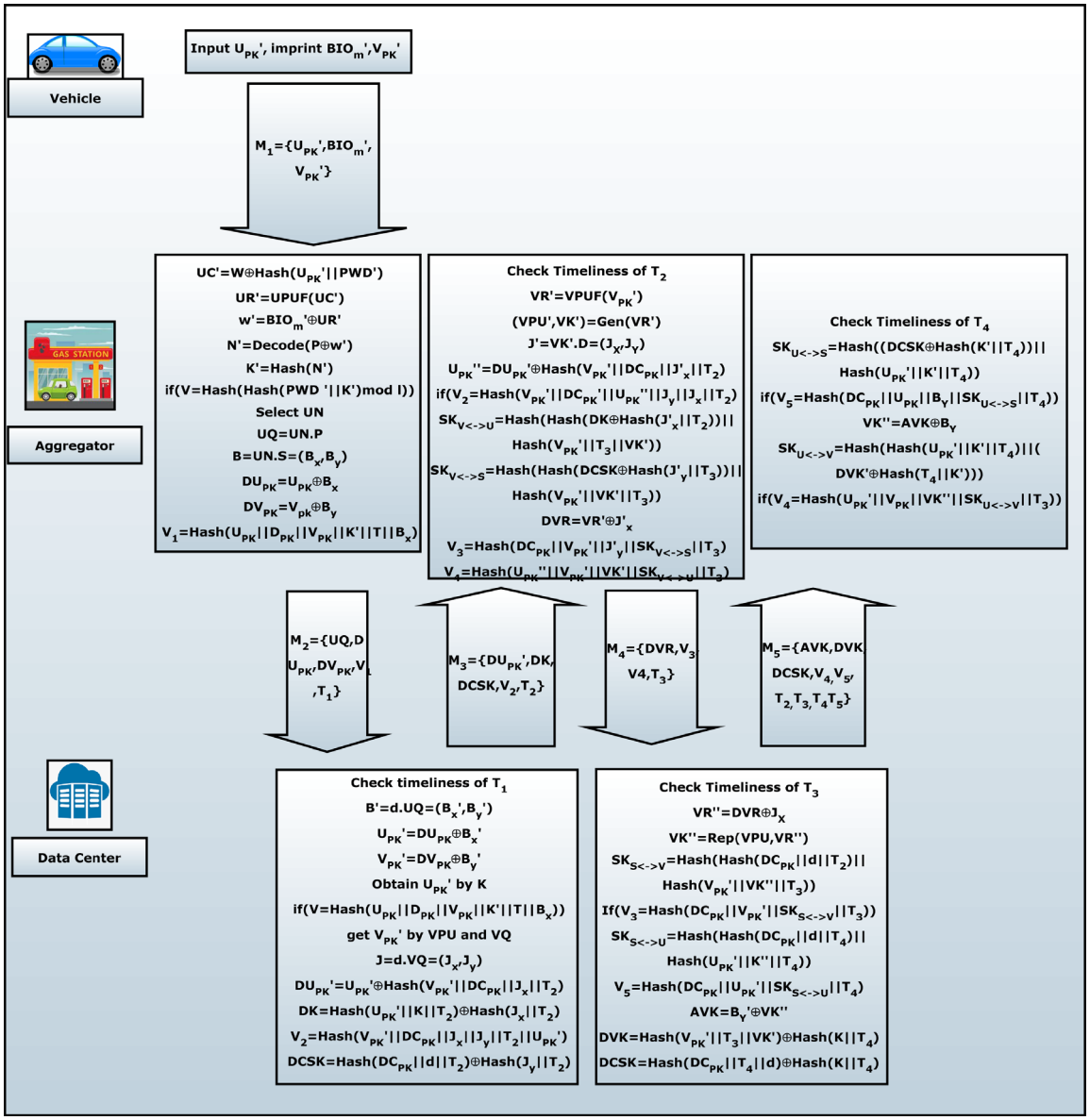

FIGURE 13 Login and authentication phase.

4.3.4 | Password update phase

This phase conducts the updating process of the password in case if the user requires to change the password or in case of any leakage of data or if he forgets the original password. The specific procedure is discussed in Figure 14 and steps are as follows:

Step 1: Step 1 of the update phase is the same as that of the authentication phase.
Step 2: If the above-calculated value equals V, then the user is requested to enter a new password.
Step 3: U enters the updated and new PWD_{new}.
Step 4: $W_{new} = UC' Hash(U'_{PK} \| PWD_{new})$, $V_{new} = Hash\left(Hash\left(PWD_{new} \| K'\right) mod\ I\right)$ are computed by the device. Finally, the device substitutes V and W with V_{new} and W_{new}, respectively.

4.3.5 | Biometrics update phase

As demonstrated in Figure 15, this step allows genuine users to alter their biometry without having to communicate with the data center.

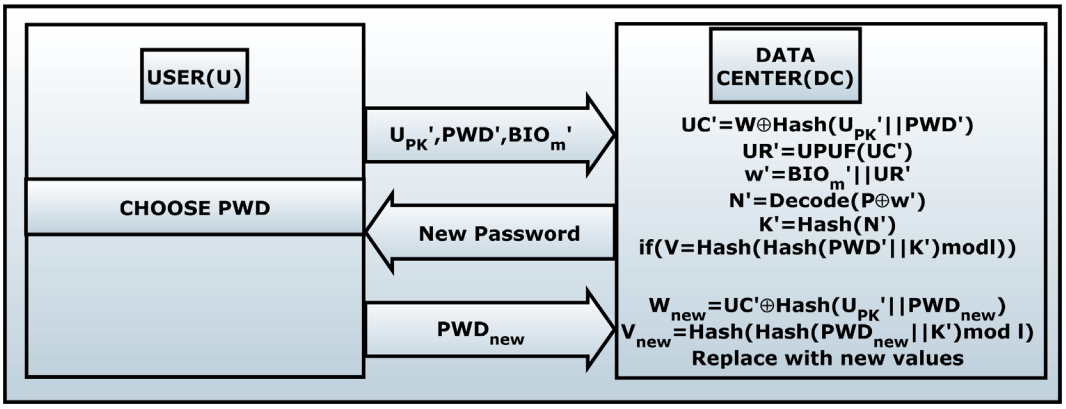

FIGURE 14 Updating password.

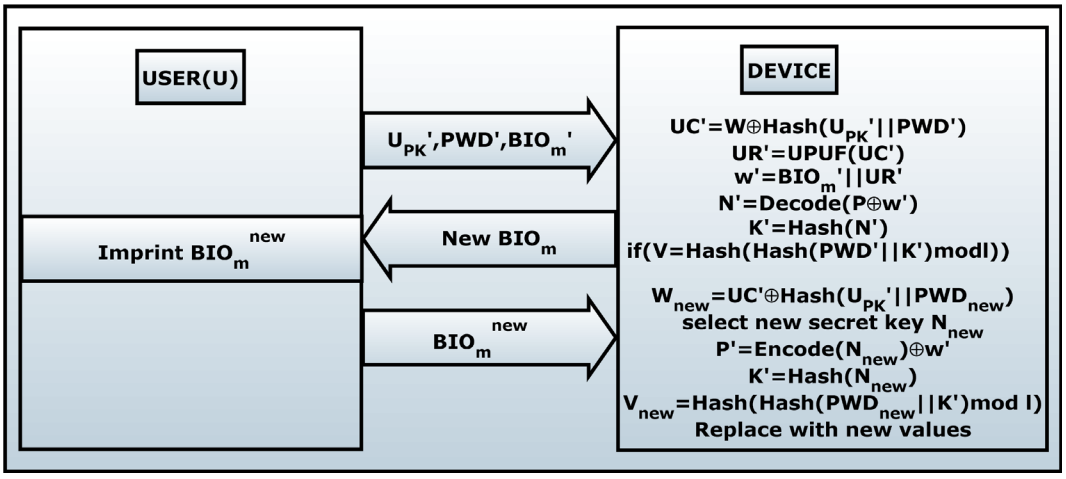

FIGURE 15 Updating biometrics.

Step 1: Step 1 of the update phase is identical to that of the authentication phase.

Step 2: If the above-calculated value equals V, then the user is requested to enter the new biometrics.

Step 3: U enters the updated and new BIO_m^{new}.

Step 4: $w' = BIO_m^{new} \oplus UR'$, $K' = Hash(N_{new})$ and $V_{new} = Hash\left(Hash\left(PWD' \parallel K'\right) \; mod \; l\right)$, is computed by using a new random value of N and then they are used to replace original values.

5 | SECURITY EVALUATION

In this section, we provide an analysis of the privacy and security formally as well as informally along with the hardware efficiency of Crossover RO PUF.

5.1 | Formal analysis

The formal analysis can be divided into three sections, that is, analysis of PUF model, analysis of shared key security and finally the security model.

5.1.1 | PUF model

PUF can be represented in mathematical form as $R = PUF(C)$, which essentially represents a relation $\{0,1\}^l \kappa 1 \rightarrow \{0,1\}^l$ where l is the length of the response and challenge fed to the PUF.

PUF responses are uniformly distributed to transform to binary strings by the fuzzy extractor. It is then tested against a probabilistic polynomial time (PPT) Adversary *A*.

Stage 1: *A* requests any challenge VC_i, inputs that to PUF and gets the response (VK_i, VPU_i).
Stage 2: *A* queries and challenges he has already queried and gets the helper data for this query.
Response: After stage 1 and 2 *A* must reply with his guess for *VK* let *VK′*. If he guesses correctly *A* wins the game but since the output of PUF is unpredictable, hence for length *l* of response the probability of *A* winning is negligible.

$$A_{win}(l) = \Pr\left[VK' = VK\right] - \frac{1}{2}.$$

5.1.2 | Analyzing the safety of the key-sharing protocol

To examine the security of the key-sharing protocol the tool was utilized and was set up on python. As shown in the figures below two attack methods were proposed to which our protocol was prone but in both of the attack flows it was assumed that the adversary had prior knowledge about the structure of the CROPUF, but it is not possible as the PUF model was only used to register and hence it resides securely in our data center without any intervention, therefore it is almost impossible to eavesdrop the messages and decode them. Figures 16 and 17 show the attack flows.

5.1.3 | Hardware efficiency

We try to compare the hardware efficiency of all ring oscillator PUFs by taking the number of Lookup tables used per bit to generate 256-bit response to a challenge as a means to compare all the efficiency. Table 4 evidently shows that Crossover PUF has outperformed all the other ring oscillator PUFs, and it is almost 11 times better than the other PUFs. The reliability threshold σ is defined as follows:

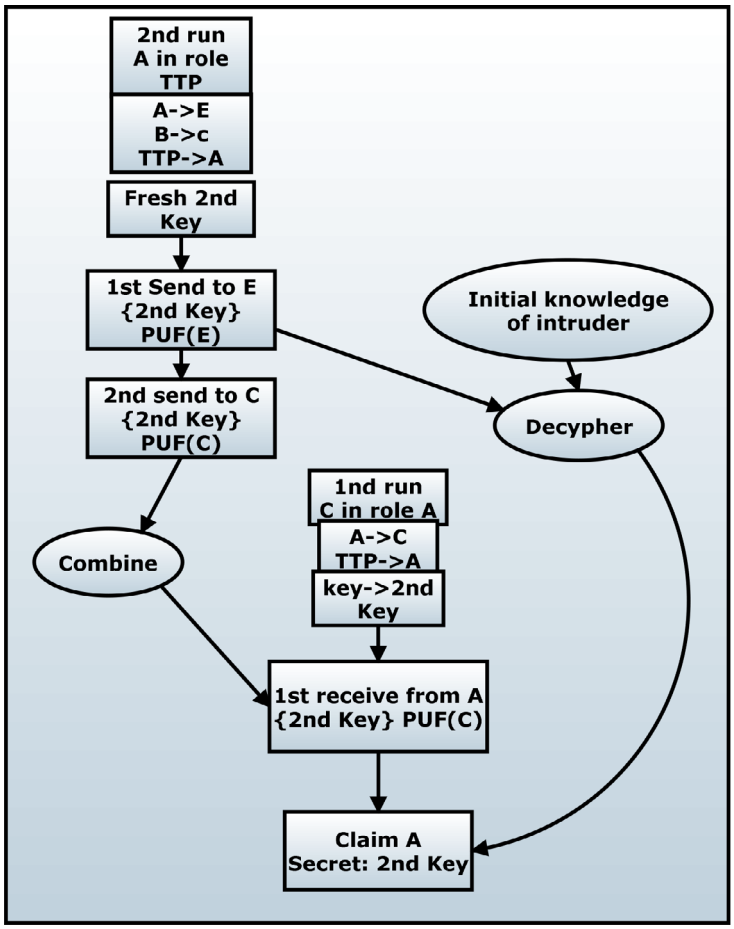

FIGURE 16 Flow of the attack method #1.

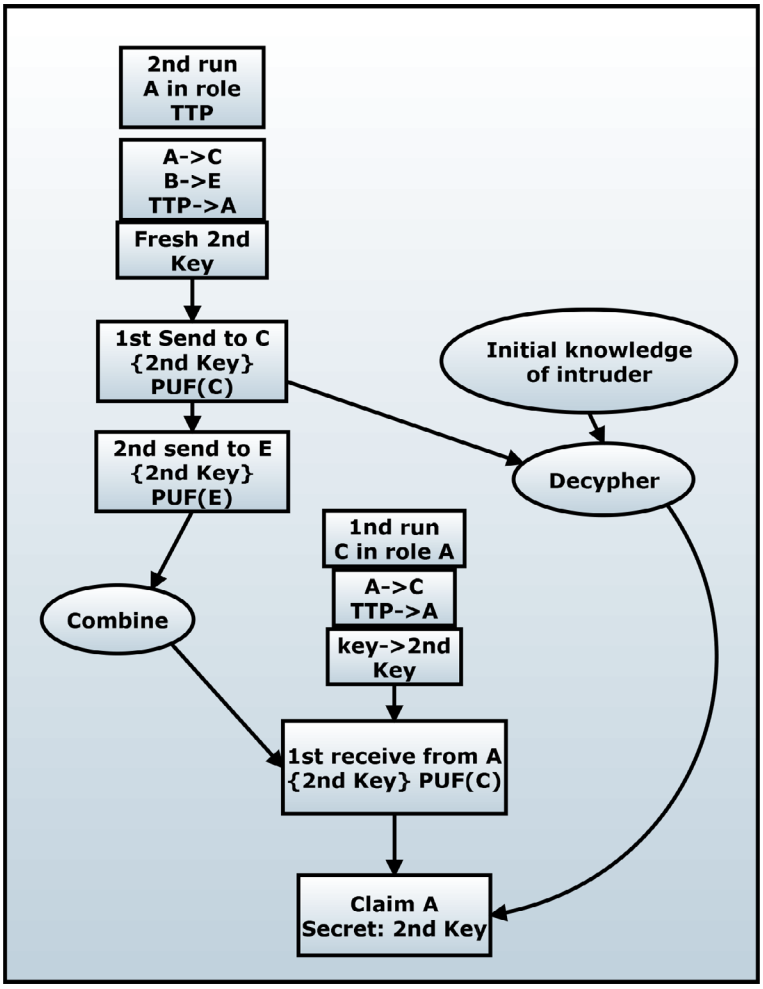

FIGURE 17 Flow of the attack method #2.

TABLE 4 Hardware overhead.

RO PUFs	Number of lookup tables
Neighbor coding	8704
rPUF	322
Crossover	62

Let frequencies of two PUFs be F_a and F_b and F_{ref} be the reference frequency, then

$$|F_a - F_b| \geq F_{TH} = F_{ref}\, \sigma.$$

As shown in Figure 18 as we make the threshold tighter the number of LUTs increase considerably, that is, more hardware resources are consumed. In Figure 18, the *x*-axis indicates the reliability threshold and *y*-axis indicates the number of LUTs per bit of conversion from challenge to response.

5.1.4 | Coefficient of stabilization

For assuring the usability of the CROPUF, we must ensure that there is substantial amount of credible challenge-response pairs satisfying our threshold condition. To ensure this we define stability of CROPUF which is the ratio of pairs that fulfill the condition versus total number of challenge

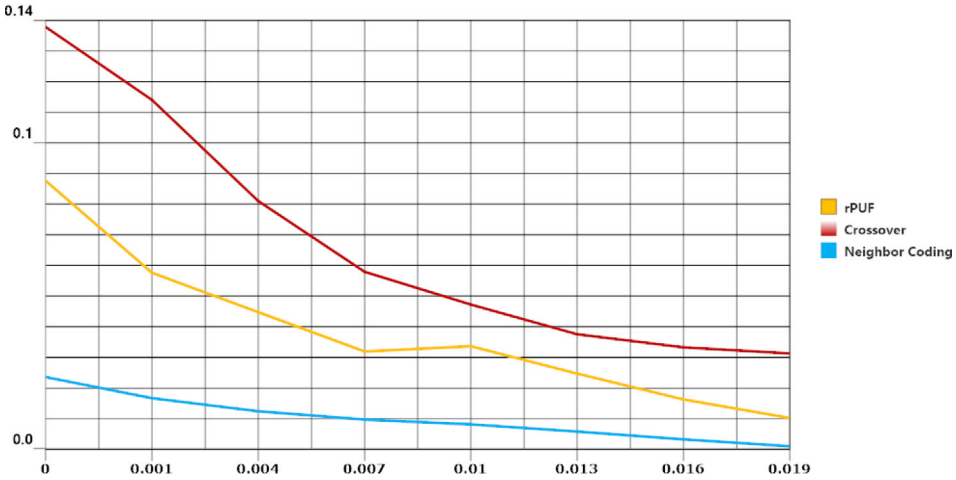

FIGURE 18 Comparison of hardware efficiency.

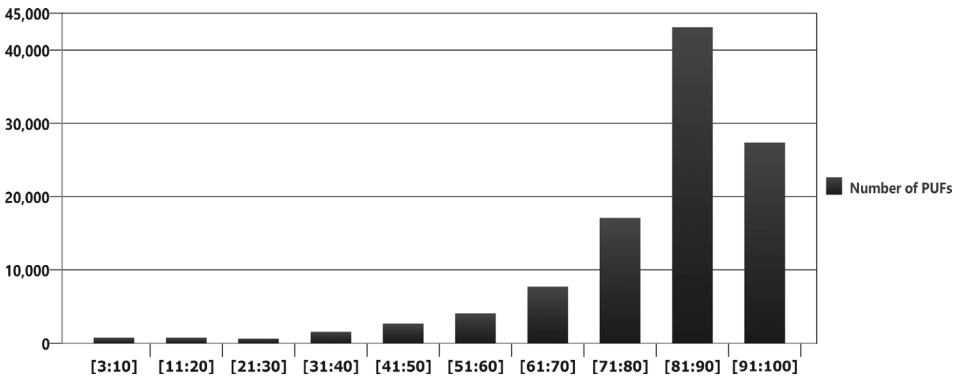

FIGURE 19 Coefficient of stabilization.

response pairs. The result of the experiment conducted on about 100,000 CROPUFS each of $4 * 5$ dimensions are shown in Figure 19. The *y*-axis represents the number of PUFs, and the *x*-axis represents the stability. 10,368 distinct CRPs may be created from a $4 * 5$ PUF.

Hence for the values of stability till 60% the number of PUFs is very less and we can successfully use the CROPUF.

5.1.5 | Security model

Assumptions

- Each entity E (U, DC or V) of the system can run parallel simulations of the protocol. The ith instance of E is denoted by π_E^i which returns a Boolean as the session finishes, the Boolean is used to indicate whether the string was accepted or not.
- The "Accept" state is produced and the identical session key SK is bargained only when the session IDs of π_E^i and $\pi_{E'}^j$ are equal, the output is "Accept" state and the identical session key SK is bargained. Since these two devices interact with each other during the session they are called partners. pid_E^i and $pid_{E'}^j$ represent π_E^i and $\pi_{E'}^j$'s ID of partner. Hence $pid_E^i = E'$ and $pid_{E'}^j = E$.
- A is the probabilistic polynomial-time (PPT) adversary who can eavesdrop, intercept, manipulate, or insert messages on the public channel among the players in P.
- At the start of every session our adversary A tries to get hold of P's session key, the only command he can run in the beginning is test.

Fresh instance

It refers to an instance in which any of the entities are not completely violated and our adversary has absolutely no information about the current session key of the instance. However, the adversary may have the previous session keys information (Algorithm 5).

Algorithm 5. Session key security

Input: set of users U, a data centre DC, vehicle set V, every user $u \in U$ has a tuple $T = \{PWD, UPUF, BIO_m\}$. Each vehicle $v \in V$ has a single feature F = $\{VPUF\}$ in its on-board sensors. DC keeps $item_U = \{U_{PK}, K\}$ for U and $item_V = \{V_{PK}, VPU, VQ\}$ for V. P signifies the three-factor protocol introduced in the manuscript.
Output: Winner of Game
Queries performed by A:
Execute () {
$Array\ M \leftarrow Eavesdrop\ (A, P)$; // While P is running our adversary will listen in on the conversations and get hold of any messages being relayed. (Passive attack)
Returns: array $M = \{M1, M2, M3, M4\}$
}
Send (π_E^i, m) {
If $(m\ is\ valid\ message)$ {
Returns: Result returned by π_E^i //Query is executed
}
Else {
Returns: Rejected //terminates execution of P
}
End if
}
Reveal (π_E^i) {
Returns: session key SK held by the instance π_E^i.
}
Corrupt (π_E^i) {
$var\ creds \leftarrow getCredentials(\pi_E^i)$\{
Returns: the credentials of the corrupted party
}
If $(E = DC)$\{
Returns: s (Secret value of DC), and verification items $item_U$ and $item_V$;
}
Else if $(E = V)$\{
Returns: $NULL$;
}
Else {
$var\ factor1, factor2 \leftarrow selectTwo(F)$\{
Returns: Two factors out of tuple F
}
If($factor1 = PWD\ and\ factor2 = UPUF$)\{
Corrupt $(\pi_U^i, PWD, UPUF)$
}
Else If($factor1 = PWD\ and\ factor2 = BIO_m$)\{
Corrupt (π_U^i, PWD, BIO_m)
}
Else If($factor1 = UPUF and\ factor2 = BIO_m$)\{
Corrupt $(\pi_U^i, UPUF, BIO_m)$
}
End if
}
End if
Test (π_E^i) {
$b \leftarrow select\ random(0,1)$
If $(b = 0)$ {
Returns: $SK\ of\ \pi_E^i$
}
Else {
$sessionKeyLength = lengthof(SK\ of\ \pi_E^i)$
$Var\ rand \leftarrow random\ string\ of\ sessionKeyLength$
Returns: $rand$
}
End if
}
Guess () {
$Boolean\ guess \leftarrow Guess\ of\ A$ //A guesses whether the result of the test is SK of π_E^i or not in polynomial time
$string\ result =$**Test** (π_E^i)
If $(result = SK\ and\ guess = True)$\{
Result: A wins
}
Else if $(result \neq SK\ and\ guess = False)$\{
Result: A wins
}
Else {
Result: P wins
}
End if
}

Definition 1. Let *Success* indicate the event that A can correctly guess the shared security key or session key ($k' = k$, the event of A winning over the model can be represented as:

$$A_{win} = 2 * Pr[Success] - 1.$$

If A_{win} satisfies the following equation the model is said to be semantically secure.

$$A_{win} \leq a \cdot n_{send} / \mid N \mid + e,$$

where, e is a small constant (negligible), a is another constant, n_{send} is the attack numbers that the attacker launches against our system, $\mid N \mid$ the number of entries in our password space.

5.1.6 | Security theorem validation

Theorem. Assuming elliptic-curve computational Diffie–Hellman (ECCDH), the event of winning for any adversary would be negligibly greater than $a \cdot n_{send} / \mid N \mid$

$$A_{win} \leq a \cdot n_{send} / \mid N \mid + neglible(l),$$

where, $neglible(l)$ represents a negligible function of l, where l is the length of challenge and response.

Proof. The stages S_i are used to demonstrate the semantic security. Here S_0 reflects the real-world setting in which A operates. The game is finished when the advantage of the adversary is zero, and this is achieved by frequently adjusting the simulation rules in the future stages. A is a PPT active attacker with n_{send} active attacks.

In each stage $A_{win}(i)$ represents the event of A securing stage and Δ_i represents the difference between two successive stages

$$\Delta_i = |A_{win}(i+1) - A_{win}(i)|.$$

Stage S_0: It is a real-world protocol that is running under the random oracle concept (ROM). A's success in this can be represented as:

$$A_{win}(0) = A_{win}.$$

Stage S_1: We duplicate hash oracles in this game by maintaining a list of all the Hashes L. A runs the query using the input x on PUF. The simulated oracle then checks list L to see if x and its matching result y are there. Oracle returns y if it is. If not, the simulated oracle returns z and chooses a string $z = \{0,1\}^l$. The item (x, z) is then stored in L by the simulated oracle. The simulation would be over in polynomial time since z is always created in polynomial time. S_0 and S_1 are indistinguishable to A. Therefore, the difference can be represented as:

$$\Delta_0 = |A_{win}(1) - A_{win}(0)| \leq neglible(l).$$

Stage S_2: Simulations for the hash oracles in S_1 have been carried out. Collisions can happen, and if they do, the game can be won by repeating the process. Because of hash function yield is sufficiently lengthy and the search number is restricted to polynomial, likelihood of a collision is small, according to the birthday paradox.

$$\Delta_1 = |A_{win}(2) - A_{win}(1)| \leq neglible(l).$$

Stage S_3: In stage 3, we use Execute () query to replicate A's ability of passive attack. Even if A obtains all the messages communicated, A cannot deliver right communications to end the game as they contain a timestamp that needs to be verified by the receiving party. Hence stages 3 and 2 are indistinguishable for A.

$$\Delta_2 = |A_{win}(3) - A_{win}(2)| \leq neglible(l).$$

Stage S_4: In stage 4 as well, we use Execute () query to replicate *A*'s ability of passive attack on *M*1. To generate the first verification message *A* need the secret response of users PUF, biometrics, and temporary key *B* as well which cannot be obtained, hence the probability to guess all three simultaneously is negligible.

$$\Delta_3 = |A_{win}(4) - A_{win}(3)| \leq neglible(l).$$

Stage S_5: In elliptic curve cryptography, the likelihood of successfully creating *V*1 is similar to the likelihood of resolving the ECCDH issue (ECC). ECCDH is a tough issue, and it is ECC's safety assurance, hence its chances of being solved are slim. As a result, the chances of *A* succeeding are nil. So, the difference would be:

$$\Delta_4 = |A_{win}(5) - A_{win}(4)| \leq neglible(l).$$

Stage S_6: For this stage, we exploit the *Send*() command and replicate the active attack on the Data center using a third verification message. *A* constructs V_3 on his own and by assuming the right PUF response, but the probability of that is negligible.

$$\Delta_5 = |A_{win}(6) - A_{win}(5)| \leq neglible(l).$$

Stage S_7: In this stage as well, we try to exploit *A*'s active attack by executing *Send*() on the user itself, but since *A* has not corrupted either of the parties, that is, data center and user, he cannot effectively send verification messages and hence the stages are not distinguishable.

$$\Delta_6 = |A_{win}(7) - A_{win}(6)| \leq neglible(l).$$

Stage S_8: In this stage, we assume that somehow *A* can get hold of session keys shared between server and user or vehicle but to generate a valid verification message *A* need to get hold of temporary keys as well but the probability of him obtaining the keys is negligible, as *A* needs to guess the value of keys. Hence the corresponding difference is shown as:

$$\Delta_7 = |A_{win}(8) - A_{win}(7)| \leq neglible(l).$$

In the end, *A* can only retrieve irrelevant *PWD* messages by using the *Execute*() query, thus having no benefit. As a result, *A* can only get information through using *Send*()

$$A_{win} \leq a \cdot n_{send} / \mid N \mid + neglible(l).$$

Hence theorem is proved. ∎

5.1.7 | Formal analysis using BAN logic

We demonstrate the provability of our proposed protocol by using a popular formal analysis tool called BAN logic.[38] The BAN logic is elaborated in the References 38, 40. The proposed network model comprises of *TA* which is entrusted by both V_i and R_j. Therefore, we can say that if both V_i and R_j share a session key with *TA* then, it is obvious that both entities mutually share a session key amongst themselves. Moreover, the authentication process takes place on the basis of implicitly shared information between V_i and *TA*. The implicitly shared information is $H(x_{TA} \| VID_{V_i})$ and $H(ID_{V_i} \| x_{TA})$ and the authentication process between R_j and *TA* takes place on the basis of shared secret key K_j. Furthermore, the *TA* generates the session key $Session_{key}$ on the basis of arbitrary numbers $a_{r_{V_i}}$ and $a_{r_{R_j}}$. These arbitrary numbers can be recovered by the secret information of V_i and R_j. Therefore, the primary aim of our protocol is as follows:

$$\text{Aim 1} : TA| \equiv V_i| \equiv \left(V_i \xleftrightarrow{r_{V_i}} TA\right),$$

$$\text{Aim 2} : TA \mid\equiv \left(V_i \xleftrightarrow{r_{V_i}} TA\right),$$

$$\text{Aim 3}: TA|\equiv R_j| \equiv \left(R_j \xleftrightarrow{r_{R_j}} TA\right),$$

$$\text{Aim 4}: TA \mid\equiv \left(R_j \xleftrightarrow{r_{R_j}} TA\right),$$

$$\text{Aim 5}: R_j| \equiv TA| \equiv \left(R_j \xleftrightarrow{Session_{key}} TA\right),$$

$$\text{Aim 6}: R_j \mid\equiv \left(R_j \xleftrightarrow{Session_{key}} TA\right),$$

$$\text{Aim 7}: V_i| \equiv TA| \equiv \left(V_i \xleftrightarrow{Session_{key}} TA\right),$$

$$\text{Aim 8}: V_i \mid\equiv \left(V_i \xleftrightarrow{Session_{key}} TA\right).$$

The idealization of the messages transmitted between both the entities V_i and R_j are idealized below for the formal analysis:

$$Mess1.V_i \rightarrow TA: \left\langle B1, \left(V_i \xleftrightarrow{r_{V_i}} TA\right)_{H\left(ID_{V_i}\|x_{TA}\right)}, B3, t_{V_i}, VID_{V_i} \right\rangle,$$

$$Mess2.R_j \rightarrow TA: \left\langle R_j, \left(V_i \xleftrightarrow{r_{R_j}} TA\right)_{K_j}, C2, ID_{R_j}\ t_{R_j}, \right\rangle,$$

$$Mess3.TA \rightarrow R_j: \left(R_j \xleftrightarrow{Session_{key}} TA\right)_{H\left(r_{R_j}||K_j||t_{TA}\right)}, D2, t_{TA},$$

$$Mess4.TA \rightarrow V_i: \left(V_i \xleftrightarrow{Session_{key}} TA\right)_{H\left(r_{V_i}||t_{TA}|IID_{V_i}\right)}, D4, D5, D6, t_{TA}.$$

Moreover, we also have certain other assumptions according to the proposed protocol:

$$A1: TA \mid\equiv V_i \xleftrightarrow{H\left(ID_{V_i}\|x_{TA}\right)} TA,$$

$$A2: TA \mid\equiv \#\left(t_{V_i}\right),$$

$$A3: V_i \mid=> \left(V_i \xleftrightarrow{r_{V_i}} TA\right),$$

$$A4: TA \mid\equiv \left(R_j \xleftrightarrow{K_j} TA\right),$$

$$A5: TA \mid\equiv \#\left(t_{R_j}\right),$$

$$A6: TA \mid\equiv R_j \mid=> \left(R_j \xleftrightarrow{r_{R_j}} TA\right),$$

$$A7: R_j \mid\equiv \left(R_j \xleftrightarrow{H\left(r_{R_j}||K_j||t_{TA}\right)} TA\right)R_j,$$

$$A8: R_j \mid\equiv \#(t_{TA}),$$

$$A9: R_j \mid\equiv TA => \left(R_j \xleftrightarrow{Session_{key}} TA\right),$$

$$A10: V_i \mid\equiv \left(V_i \xleftrightarrow{H\left(r_{V_i}||t_{TA}||ID_{V_i}\right)} TA\right),$$

$$A11: V_i \mid\equiv \#(t_{TA}),$$

$$A12: V_i \mid\equiv TA => \left(V_i \xleftrightarrow{Session_{key}} TA\right).$$

Now, we try to prove that our proposed protocol is achieving the described aims in accordance with the BAN logic and its respective explanation is as follows:

From *Mess1* we infer:

$$S1 : TA \leftarrow \left\langle B1, \left(V_i \xleftrightarrow{Session_{key}} TA\right)_{H\left(ID_{V_i} \| x_{TA}\right)}, B3, t_{V_i}, VID_{V_i} \right\rangle.$$

By the application of message meaning rule, from *S*1 and *A*1 we infer,

$$S2 : TA| \equiv V_i| \sim \left(V_i \xleftrightarrow{r_{V_i}} TA\right).$$

Applying the freshness rule and from *A*2, we have $TA \mid\equiv \#\left\langle B1, \left(V_i \xleftrightarrow{r_{V_i}} TA\right)_{H\left(ID_{V_i} \| x_{TA}\right)}, B3, t_{V_i}, \ VID_{V_i} \right\rangle$ and because of *A*1 we infer:

$$S3 : TA \mid\equiv \#\left(V_i \xleftrightarrow{r_{V_i}} TA\right).$$

By implementing nonce verification and from *S*2 and *S*3 we deduce:

$$S4 : TA| \equiv V_i| \equiv \left(V_i \xleftrightarrow{r_{V_i}} TA\right) \text{ Aim 1.}$$

By implementing jurisdiction rule and from *A*3 and *S*4 we infer:

$$S5 : TA \mid\equiv \left(V_i \xleftrightarrow{r_{V_i}} TA\right).$$

From *M*2 message we comprehend:

$$S6 : TA \leftarrow \left\langle \left(R_j \xleftrightarrow{r_{R_j}} TA\right)_{k_j}, C2, ID_{R_j}, t_{R_j} \right\rangle.$$

By using message meaning rule and from *S*6 and *A*4, we infer:

$$S7 : TA| \equiv R_j| \sim \left(R_j \xleftrightarrow{r_{R_j}} TA\right).$$

Using the freshness rule and by *A*5, we have $TA \mid\equiv \#\left\langle \left(R_j \xleftrightarrow{r_{R_j}} TA\right)_{k_j}, C2, ID_{R_j}, t_{R_j} \right\rangle$, and because of *A*4 we infer:

$$S8 : TA \mid\equiv \#\left(R_j \xleftrightarrow{r_{R_j}} TA\right).$$

Again, implementing nonce verification rule and from *S*7 and *S*8 we infer:

$$S9 : TA \mid\equiv R_j \mid\equiv \left(R_j \xleftrightarrow{r_{R_j}} TA\right) \text{ Aim 3.}$$

By implementing jurisdiction rule and from *A*6 and *S*9 we infer:

$$S10 : TA \mid\equiv \left(R_j \xleftrightarrow{r_{R_j}} TA\right) \text{ Aim 4.}$$

From *M*3 message we comprehend:

$$S11 : R_j \leftarrow \left\langle \left(R_j \xleftrightarrow{Session_{key}} TA\right)_{H\left(r_{R_j} | K_j | t_{TA}\right)}, D2, t_{TA} \right\rangle.$$

By using message meaning rule and from $S11$ and $A7$, we infer:

$$S12: R_j| \equiv TA|\sim\left(R_j \xleftrightarrow{Session_{key}} TA\right).$$

Using the freshness rule and by $A8$, we have $R_j| \equiv TA|\sim\left(R_j \xleftrightarrow{Session_{key}} TA\right)_{H\left(r_{R_j}||K_j||t_{TA}\right)}, D2, t_{TA}$ and because of $A7$ we infer:

$$S13: R_j| \equiv \#\left(R_j \xleftrightarrow{Session_{key}} TA\right).$$

Again, implementing nonce verification rule and from $S12$ and $S13$ we infer:

$$S14: R_j| \equiv TA| \equiv \left(R_j \xleftrightarrow{Session_{key}} TA\right).$$

By implementing jurisdiction rule and from $A9$ and $S14$ we infer:

$$S15: R_j| \equiv \#\left(R_j \xleftrightarrow{Session_{key}} TA\right).$$

From $M4$ message we comprehend:

$$S16: V_i \leftarrow \left\langle \left(V_i \xleftrightarrow{Session_{key}} TA\right)_{H\left(r_{V_i}||t_{TA}||ID_{V_i}\right)}, D4, D5, D6, t_{TA} \right\rangle.$$

By using message meaning rule and from $S16$ and $A10$, we infer:

$$S17: V_i| \equiv TA|\sim\left(V_i \xleftrightarrow{Session_{key}} TA\right).$$

Using the freshness rule and by $A11$, we have $TA| \equiv \#\left\langle \left(V_i \xleftrightarrow{Session_{key}} TA\right)_{H\left(r_{V_i}||t_{TA}||ID_{V_i}\right)}, D4, D5, D6, t_{TA} \right\rangle$ and because of $A10$ we infer:

$$S18: TA| \equiv \#\left(V_i \xleftrightarrow{r_{V_i}} TA\right).$$

Again, implementing nonce verification rule and from $S17$ and $S18$ we infer:

$$S19: V_i| \equiv TA| \equiv \left(V_i \xleftrightarrow{Session_{key}} TA\right) \text{ Aim 7.}$$

By implementing jurisdiction rule and from $A12$ and $S19$ we infer:

$$S20: V_i| \equiv \left(V_i \xleftrightarrow{Session_{key}} TA\right) \text{ Aim 8.}$$

Hence, we can see that by using the above formal security analysis our proposed protocol has achieved all the eight security goals and also the entities V_i and R_j have also been successful in sharing a session key with the help of TA.

5.2 | Informal analysis

1. Perpetrator A possesses complete control over the communication channel between the communicating parties and can launch both passive and aggressive attacks against it. A, for example, has the ability to listen in on, remove, and change any communication sent over the public link.
2. In polynomial time, all the possible candidates in the Cartesian product are computed by A of the domain of user identification and user password domain.

3. During the assessment of security aspects such as security of session key and potential adversaries like impersonation attacks *A* must be aware of the victim's identity.
4. *A* can access two of the three factors involved in authentication and hence we can categorize the outcomes into three parts:
    1. *A* accesses the biometrics along with the user device.
    2. *A* accesses the biometrics along with the user's password.
    3. *A* accesses the user's password along with the user device.

In the following sections we present a detailed description of infeasibility of several attacks against our proposed protocol.

5.2.1 | Attacks which target the issue of de-synchronization

If an adversary *A* intercepts and is able to access the messages three and four, the protocol's three participants, *U*, *DC*, and *VS*, are unable to verify the identity of each other and hence fail to gain a shared session key, and so the protocol is unable to function correctly. Because each user's/identification vehicle is solely recorded in *DC*'s database, there is no way to change it throughout the authentication process. Users' passwords and biometrics are changed on a per-user basis, without the need for *DC* interaction. Even if *A* intercepts these signals, the local data cannot be incompatible with the *U*'s database in the DC's database. As a result, de-synchronization attacks are not possible using the suggested protocol.

5.2.2 | Avoiding attacks of impersonation

- *Impersonation attacks as U*: To impersonate as the user the adversary will need access to all of the three factors simultaneously which is highly unlikely the paper already discussed the case when the adversary is able to gain access to any two out of these three factors, but without the three factors combined the adversary will fail the verification to *DC* and will not be able to generate appropriate session key to communicate over the server. In the first message itself all of the three factors are necessary to be validated on the data center.
- *Impersonation attacks as V S*: If adversary tries to impersonate as *VS*, it will try to access the PUF and private key of vehicle device, therefore, without having the device access it cannot generate suitable response even after having the corresponding challenge. In addition, without the private key of *VS* its impossible to generate the session keys and get verified to the data center.

Impersonation attacks as DC: Let us assume that adversary is somehow able to get access to the entire database of data center, that is, it can gain access to all the public/private keys of users/vehicles stored but not data center's own private key. Thus, without having any knowledge of *d*, it is not possible to compute the temporary key *B*, without which the private keys and public keys are totally useless. It can never generate the essential session keys and thus cannot differentiate one key from another. Similarly for vehicle verification/login as well the temporary key cannot be generated accurately without the private key and thus the adversary cannot obtain the appropriate session keys or the public/private keys of corresponding devices.

5.2.3 | Avoiding attacks of modification

Considering the first message, that is, M_1 was tempered by the adversary and updated to some false values, after receiving the message the data center calculates its temporary key, *B* using its private key *d* then data center recovers the private key of user. Adversary can select a random number in place of the temporary key but it will fail to verify the message as the adversary has no information about the private key of user, which is registered in the data center, thus it will fail in the verification step and will get rejected and eventually terminated.

Now for the message M_2, the adversary has no way of accessing the data centers private key *d*, thus he cannot generate the temporary key, or the message thus generated would be invalid and will be rejected.

For M_3 the adversary needs to compute V_3 which is impossible as from the analysis above to obtain V_3 the adversary must have the temporary key *J* which needs to be extracted from M_2 which is also not possible. Hence, he cannot get private keys of data centet and user and thus cannot construct the necessary session key. Thus, the message generated by adversary would be invalid.

Now let us assume the adversary tampers the message M_4 but since the next verification step, that is, V_4 contains the session key between vehicle sensor and user and we know that adversary can never get hold of this, we can be sure that even after tampering he/she cannot get access to the device. Hence, all of our messages are resistant to attacks of modification.

5.2.4 | Avoiding attacks of replay

Since timestamp is an integrated feature in almost all of the verification messages, if an adversary were to replay the messages to send fraudulent messages after some time it would not be possible. Each system component confirms the of the received message timestamp. In addition, the timestamp involved in the message do not differ by a bigger value, that is, they are near to each other. If the values differ by a great margin the message will be rejected, and session would be terminated. Only after the verification of timestamp, the message is actually verified against the information stored on the device.

5.2.5 | Avoiding attacks of offline password guessing

To obtain the credentials such as password of the adversary must decode it from the message $Hash(PWi \bigoplus UKi) mod I$ or from the message $W = UC \bigoplus Hash(U_{PK} \bigoplus PWD)$, and since the adversary can never get hold of all the three factors simultaneously, he/she can never get the private key of the user, since it is also dependent on the PUF response. Since the password is also protected using the fuzzy verifier it is almost impossible for the adversary to correctly guess the password in such a huge space of all passwords.

5.2.6 | Mutual authentication

There are two major communication channels where mutual authentication is required, the first one is between *DC* and *U*. To ensure the same a random number string is chosen by *U*. *U* then calculates a temporary key by using *DC*'s public key $B = UNi \cdot S = (Bx, By)$. This temporary key is used to encrypt the shared messages. On the data center side, *DC* uses its secure value *d* to obtain $B = s \cdot UQi$ to decrypt the message. After obtaining all the required details the data center calculates the first verification message $V1 = Hash(UIDi \parallel IDs \parallel V\ IDk \parallel UKi \parallel T1 \parallel Bx)$. Since the value of d is private and it is only known to the data center and user, the user can verify the identity of *DC* using it and generate the next message $V5 = Hashh(IDs \parallel UIDi' \parallel By \parallel SKU - S \parallel T4)$. After verifying the above mentioned messages the identities of both the data center and user are confirmed and after that we calculate our first session key between user and system: $h(h(IDs \parallel T4 \parallel d) \parallel h(UIDi \parallel UKi \parallel T4)))$ ensuring mutual authentication between the two.

Next comes the mutual authentication of *DC* and *V S*. It is very similar to our previous authentication; The data center uses the value of private key *d* and the public key that the vehicle shares to generate a temporary key $J = s \cdot V\ Qk = (Jx, Jy)$. This key is then shared to vehicle as well. The data center then encrypts the message $Hash(IDs \parallel d \parallel T2)$. After this the data center communicates the message $V2 = Hash(V\ IDk \parallel IDs \parallel UIDi \parallel Jx \parallel Jy \parallel T2)$ to *V S* for verification. To confirm the uniqueness of the data center and to ensure that the verification message is correct the vehicle sensor creates a temporary key, which in turn decrypts the previous message getting the session key between data center and vehicle sensor. Then the verification of vehicle sensor is done by data center by validating the value of *V*3. Finally, data center generates the session key to encrypt any further messages containing private information.

To generate the final session key between user and *VS* is do not by first verifying the message *V*4 to authenticate user to the vehicle sensor. Only after it is verified the data center generates the session key for the same. The authentication of *V Sk* to *U*, it is done through *DC*. Since mutual authentication between *DC* and *V Sk* has been conducted, *V Sk* can therefore trust the validity of the user without having to check on its own.

5.2.7 | User anonymity and untraced ability

The temporary key exchanged with *DC* is used to process the IDs of *Ui* and *V Sk* in the proposed protocol. Furthermore, because each session's timestamp is different, *A* is unable to deduce the participant's identity from the sent messages. Furthermore, prior communications cannot be linked to individual people or cars.

5.2.8 | Physical security

Any alteration or damage to a device will cause the PUF to behave different from original, or the device will become unusable, according to PUF's characteristics. It is difficult to collect any relevant information in an accessible area since car sensors do not preserve any information. Physical assaults, aside from rendering the hardware components in the proposed protocol ineffective, are unable to extract any relevant information. As a result, the suggested protocol can assure the system's physical security.

6 | COMPARATIVE ANALYSIS OF PROPOSED PROTOCOL WITH RELATED PROTOCOLS

Table 5 shows the formal verification of the security analysis of CROPUF and comparison of various security features with applicable protocols are presented in the table. For instance, the query (< User, IoV, RSUi >, C) says about the adversarial system of the PUF learning-challenge response pair which are vulnerable to ML attacks. Another instance is modelling a query (< User, IoV, RSUi >) for the adversarial capability of the temporary secrets in the protocol. The forward secrecy and stolen smart card attribute are verified by the testing the resiliency of the protocol by the query (< User, IoV, RSUi >).

7 | PERFORMANCE EVALUATION

The performance of our proposed protocol is evaluated in terms of some parameters such as communication overhead, computation overhead and security characteristics.

7.1 | Computation overhead

The computation overhead is defined as combined implementation time of the cryptographic operations. To calculate overhead of computation for our proposed protocol, we have taken the following functions of cryptography:

T_{ow}: Time for implementing hash function (one-way).
T_{epm}: Time for implementing time to execute elliptic curve point multiplication.
$T_{sym\ E/D}$: Time required to implement symmetric encryption/decryption.

Each side calculates the computation overhead, that is, V_i, RSU_j, TPA_t. The process of registration is a one-time process in an authentication protocol and therefore it is dropped in comparative analysis. Hence, there are three main phases in determining computation overhead namely: login, verification and key agreement. Our protocol executes the hash function thrice and the CROPUF is implemented one time by V_i. Hence, the V_i's overhead of computation comes out to be $(3 \times 1.14) + (1 \times 0.2) \approx 3.62$ ms. Correspondingly, our proposed protocol executes the hash function three times at RSU_j, whereas encryption/decryption and hash function are implemented two times and seven times for TPA_t.

TABLE 5 State-of-the-art comparison of attack scenarios with its related protocols.

Security feature	Roman et al.[9]	Rabieh et al.[11]	Raveendra et al.[14]	Revilla et al.[15]	Raveendra et al.[16]	Proposed
Formal proof	✓	×	✓	×	✓	✓
User anonymity	✓	✓	✓	✓	✓	✓
DoS attack	×	✓	✓	×	✓	✓
Privacy of location	✓	×	✓	✓	✓	✓
Resistance to MITM attack	✓	×	✓	✓	✓	✓
Privileged insider attack	✓	✓	✓	✓	✓	✓
Resistance to impersonation attack	✓	×	✓	✓	✓	✓
Resistance to ML attack	NA	NA	NA	NA	NA	✓
Provable security	×	×	✓	×	✓	✓
Parallel session attack	NA	NA	NA	NA	NA	✓
Stolen smart card verifier attack	NA	NA	✓	NA	✓	✓
Seamless handover	×	×	×	×	✓	✓
Forward secrecy	✓	✓	✓	✓	✓	✓
Password leakage attack	NA	NA	✓	NA	✓	✓

TABLE 6 Computation overhead.

Protocol	V_i (in ms)	RSU_j (in ms)	TPA_t (in ms)	Total cost (in ms)
Ma et al.[45]	5.1	0.008	0.0107	5.215
Li et al.[46]	7.1	0.0066	0.0115	7.1181
Mohit et al.[44]	7.9	0.0150	0.0033	7.918
Proposed	3.62	0.0048	0.00594	3.630

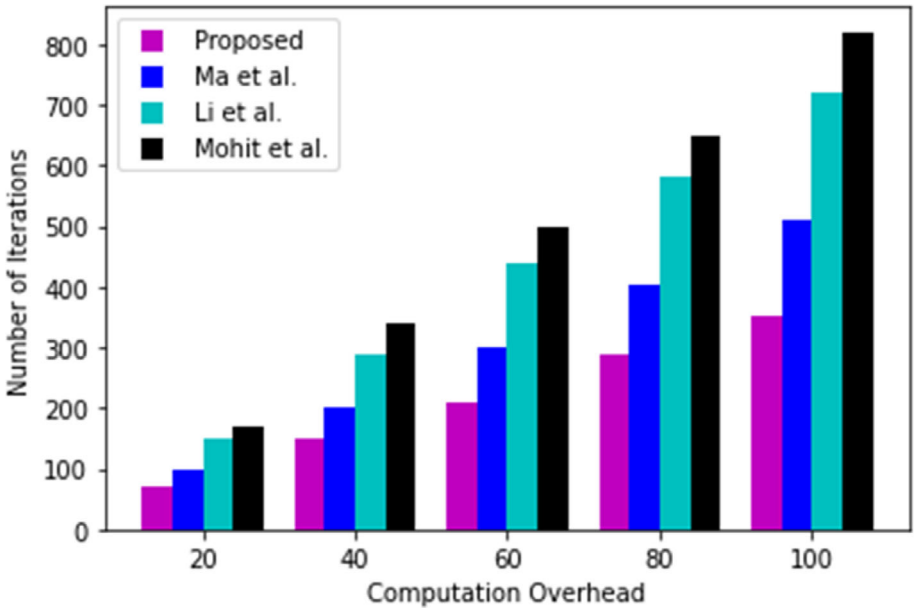

FIGURE 20 Analysis of computation overhead.

Therefore, the RSU_j and TPA_t computation overhead come out to be $3 \times 0.0016 \approx 0.0048$ and $(7 \times 0.0008) + (2 \times 0.00017) \approx 0.00594$ ms respectively. Now, the complete overhead comes out to be 3.630 ms. The overhead of computation of connected protocols[44–46] are also calculated in the equivalent way and are shown in Table 6 and Figure 20.

7.2 | Communication overhead

The communication overhead is defined as total number of bits traded amongst the partakers to finish the process of authentication. In this sub-section, we compare the relevant protocols[13–15] with our proposed protocol on the basis of communication overhead. The registration stage is an initial process and consequently, we compute the communication overhead of verification messages and the stage of key exchange. For computing the same, we take the identity size, an arbitrary number, elliptic curve point, password, XOR operation, and a quantum of 160 bits each. In case of AES-128 the size of plain-text/ciphertext is 128 bits. Contrasting to which, the hash function's digest value takes 256 bits. Four messages are exchanged between the entities V_i, RSU_j, TPA_t in our protocol. The proposed protocol and the other related protocols communication overhead is determined in the Figure 21 and Table 7.

8 | SIMULATION RESULTS

We conducted our simulation on the NS-3 platform which is a vehicle simulator. It facilitates different models to experiment differently. There are various libraries utilized in NS-3 which enables different margins and operations to support varied networks.

Table 8 displays the fundamental requirements for the simulation of our proposed protocol. The mobility of vehicles is in randomized fashion and the inter-vehicular distance is 20 m. There is an increment of 20 step-size which reaches to 100 vehicles. The scenario comprises one *TA*, numerous

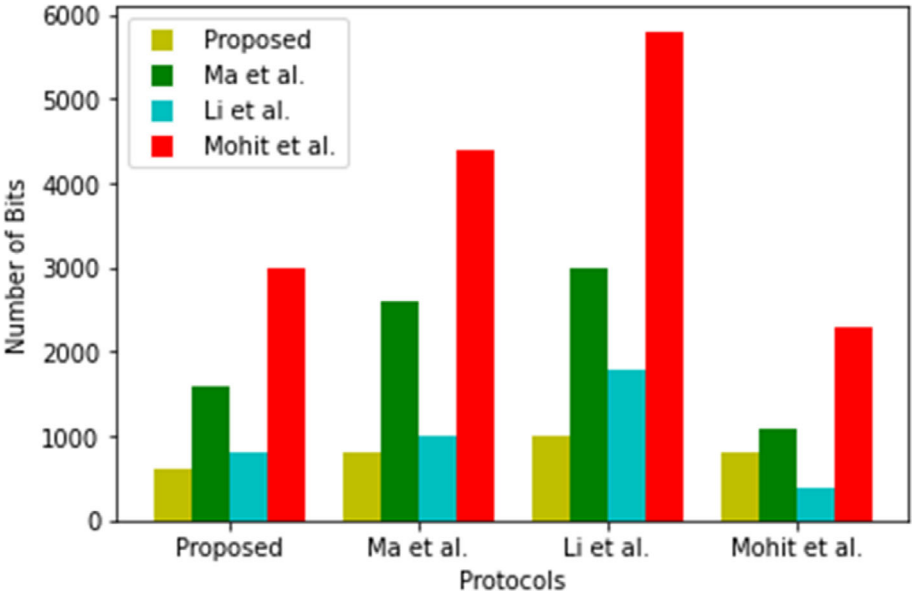

FIGURE 21 Analysis of communication overhead.

TABLE 7 Communication overhead.

Protocol	V_i	RSU_j	TPA_t	Total cost
Ma et al.	600	1600	800	3000
Li et al.	800	2600	1000	4400
Mohit et al.	1000	3000	1800	5800
Proposed	800	1100	400	2300

TABLE 8 NS-3 simulation parameters.

Parameter	Value
NOV	100
OS	Ubuntu 20.04
Geographical distribution	15 km * 15 km
Mobility model	RandomWalk2dMobilityModel
NS-3 version	3.32
Bandwidth	2 Mbps
Type of channel	Wireless
Size of packet	512 bytes
Time for simulation	1700 s

RSUs, V_i and according to the message communication cost the costs are 576, 1152, 832, and 416 bits, respectively for D1, D2, D3, and D4. The user of the vehicle transmits a packet after every 2 s. Therefore, we gauge the performance of our proposed protocol by testing three performance metrics namely: throughput, end-to-end delay and packet delivery ratio. The results of the same are shown in the Figure 22.

9 | RESULTS AND ANALYSIS

Table 9 describes the security attributes as compared to other authors and the various features of security are discussed below in detail.

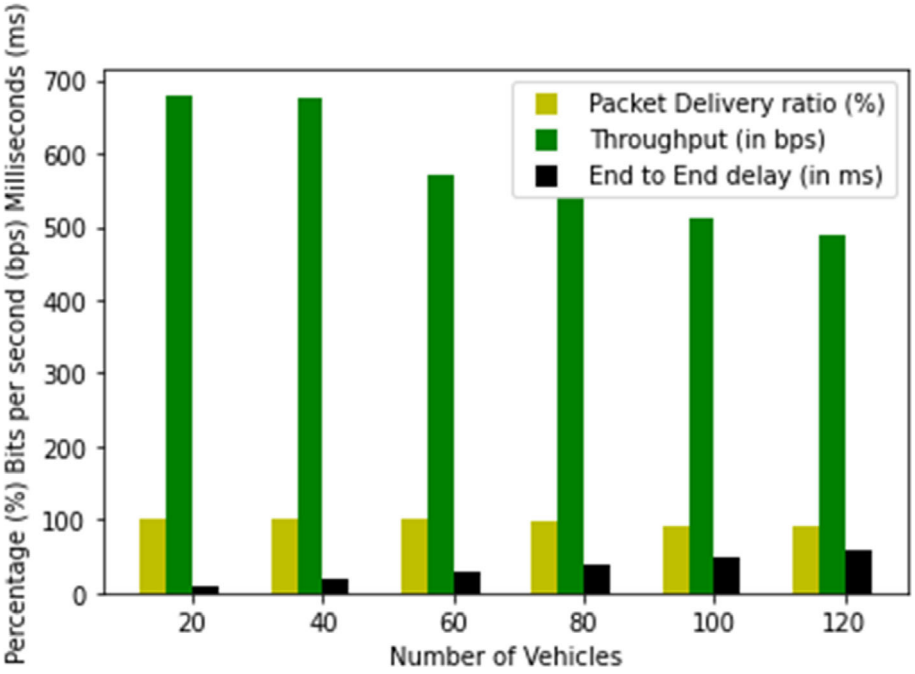

FIGURE 22 Parametric measurements for protocol performance measurement.

TABLE 9 Comparison of features of security with other authors.

Security feature	Islam et al.[61]	Cui et al.[62]	Proposed
Vehicle anonymity	✓	✓	✓
Traceability	✓	✓	✓
Non-link-ability	×	×	✓
Mutual authentication	×	×	✓
Forgery attack resistance	✓	✓	✓
Modification attack resistance	✓	✓	✓
Replay attack resistance	✓	✓	✓
Implement-ability for real application	×	×	✓

9.1 | Vehicle anonymity

This security feature is an important aspect in the protection of vehicle's privacy as it guarantees the real identities of the vehicles are safe and is not recognizable by any adversaries. In the phase of message authentication, $B2$ of the V_i's message $M1 = \{B1, B2, B3, t_{V_i}, VID_{V_i}\}$ contains the real identity ID_{V_i} of the vehicle, where the $B2 = A2\ XOR\ r_{V_i}$ and $A2 = H(ID_{V_i}||x_{TA})$ are computed in accordance with secret key x_{TA} of TA. As $A2$ is contained in a tamper-proof component and TA holds only x_{TA}, the adversary A cannot get hold of $A2$ or x_{TA} for the calculation of real identity ID_{V_i} and therefore, the vehicle anonymity attribute is satisfied.

9.2 | Non-linkability

This feature does not allow any attacker to link different sessions of a vehicle with the help of its messages through a public link. This is a special aspect of the protection of privacy in vehicles. If we retain the dynamic changes of every login session it helps in achieving the attribute of non-linkability.

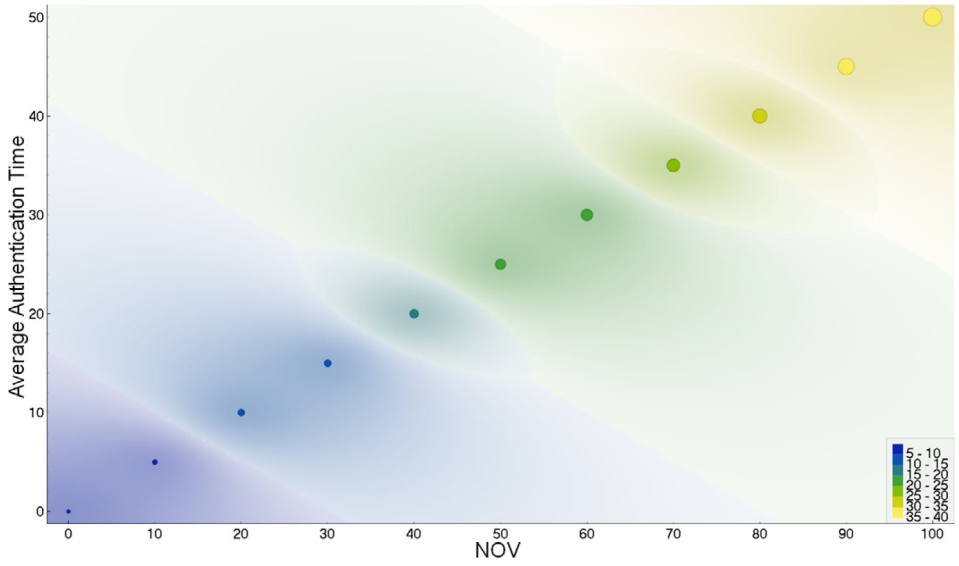

FIGURE 23 Time verification with number of vehicles (NOV).

9.3 | Traceability

This feature helps in identifying the malicious vehicles in the system by enabling the authorization entity to reveal the vehicle's real identities. As talked above, we can see that the adversary A is unable to discover the real identity of the vehicle due to the vehicle anonymity feature, however, when the message $M2 = \left\{B1, B2, B3, t_{V_i}, VID_{V_i}, C1, C2, \ ID_{R_j}, t_{R_j}\right\}$ is received in our protocol the TA can easily recover ID_{V_i} with the help of $M3$ by computing $ID_{V_i} = B1\ XOR\ H\left(H\left(x_{TA}||VID_{V_i}||t_{V_i}\right)\right)$ and hence the traceability attribute can be satisfied.

9.4 | Mutual authentication

In our proposed protocol the RSU act as a joining link between TA and vehicles as TA is a third party with which both V_i and R_j achieves mutual authentication and both V_i and R_j perceive each other valid if the other realizes mutual authentication with TA.

9.5 | Resistance to forgery attack

For the impersonation of V_i, the attacker needs to produce a valid message $M1 = \left\{B1, B2, B3, t_{V_i}, VID_{V_i}\right\}$ on behalf of V_i. However, the virtual identity of V_i is randomized after every session that is successful in our protocol and each new VID_{V_i} remains unknown to the attacker. Moreover, on the basis of $A1$ and ID_{V_i}, $B1 = H\left(A1||t_{V_i}\right)XOR\ ID_{V_i}$ in $M1$ is computed, where, the TA assigns $A1 = H\left(x_{TA}||\ ID_{V_i}\right)$ after hash functions x_{TA} and ID_{V_i} are not known to any attacker. Hence, in the absence of any required information $\left\{ID_{V_i}, VID_{V_i}, x_{TA}\ \right\}$, the attacker is unable to do any kind of forgery of a valid message $M1$ impersonating V_i.

In the above Figure 23, x-axis depicts the number of vehicles and y-axis depicts the average time taken for authentication. The scatter plot depicts four colored regions representing the proposed protocol, Mohit et al.,[44] Ma et al.,[45] Li et al.[46] observations. It displays the trend that if we increase the number of vehicles overlapping happens. It can be witnessed that the proposed mechanism does not increase much authentication time. The other Figure 24 shows the plot between vehicle moving speed versus average delay in the message transmission. We can see the delay in the message is not perturbed that much with the speed change of vehicles.

10 | DISCUSSION

There are many studies in the earlier times that have been conducted for the information systems of different domains in which public key cryptosystem is a crucial part. The authors such as Zhang et al.[11] suggested an identity-based signature scheme and revealed the first implementation of its white-box structure in the IEEE P1363 standard. This research avoided any compromise in the private security key during the execution of the

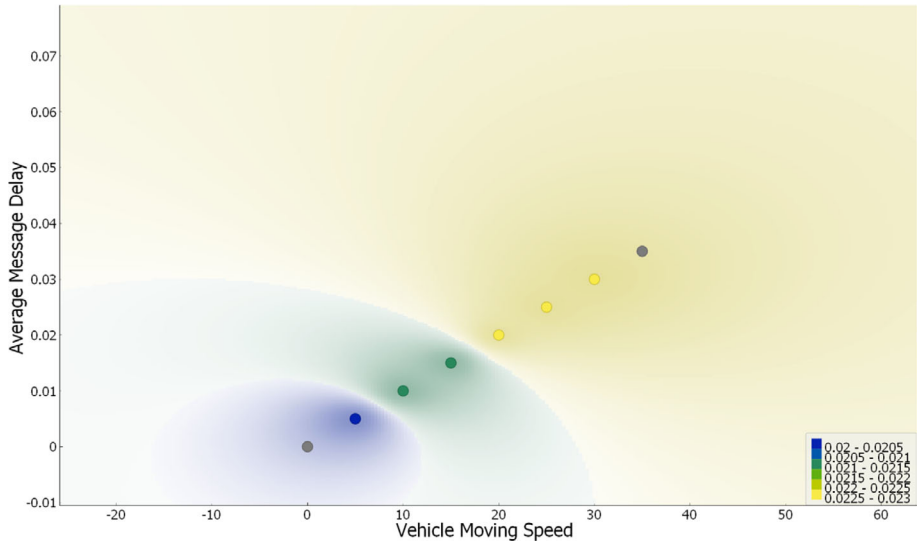

FIGURE 24 Average message delay versus speed in m/s.

algorithm in a less trustworthy environment and therefore, it could be utilized for security assurance of the mobile devices. However, in VANET since the wireless communication is the communication medium for a volatile environment of high-mobility vehicles, the communication between the vehicles becomes more vulnerable and therefore, secure transmissions mechanisms are required which should be also light in weight for its practical implementation. Moreover, authentication[12–16] is also important for the protection of OBUs from inter/intra adversaries, other malicious OBUs. Lu et al.[17] was the first person who proposed the notion of conditional privacy preservation. It means that an OBU is non-traceable by public but its real identity can be identified by TA. Further, to achieve the anonymous authentication, they proposed short-lived anonymous certificates based on which conditional privacy preservation protocol was built. However, because of frequent application for these certificates the system efficiency decreases. Zhang et al.[18] proposed a protocol which was having its authentication based on identity with conditional privacy and did not require any certificate amongst RSU and vehicles. Though, authors like Shim[19] and Lee et al.[20] drew their attention towards few weaknesses of the protocol such as absence of nonrepudiation in References 19,20 and replay attack in Reference 18. In Reference 21, Liu et al. presented that the protocol in Reference 19 can only deliver a weaker level of security than appealed. Furthermore, Zhang et al.[22] claimed that the protocol proposed in Reference 20 is prone to tracking and forgery attacks. In addition, some new improvements were proposed in References 24,25 to improve the He et al.'s[23] identity-based authentication with no bilinear pairing protocol. In state-of-the art, Islam et al.[28] suggested a novel VANET authentication protocol having a group key agreement and password while Cui et al.[29] proposed even better and efficient protocol than.[28] However, in both[28,29] for the authentication phase every vehicle has some fixed arbitrary number. This helps if the pseudo-identity of the vehicle gets changed and the changeless arbitrary number acts as an identification to the vehicle.

11 | CONCLUSION AND FUTURE WORK

Conventional cryptographic techniques have failed to provide secure communications in many IoT applications and embedded systems. The novel cyber-physical systems require lightweight security primitives. In this manuscript, we proposed a three-factor authentication system based on CROPUF that have improved the reliability of the security system. The key finding of our protocol ensures that even if the device is compromised the anonymity of the user will be maintained and no false information would be shared across the system thereby, defeating the security attacks to be able to gain access to sensitive information. Moreover, to handle the issue of noise we make use of a fuzzy extractor. The formal security analysis using BAN logic conducted in the article demonstrates that our proposed CROPUF effectually caters to the stolen smart card attack, impersonation attacks, desynchronization attacks, and also mutually authenticates the vehicular entities amongst each other provided with user anonymity attribute. Furthermore, the use of aggregators as a communication fog layer reduces latency and traffic on the communication channels. Our proposed protocol can be utilized in the real-time applicative scenario of V2V communication on the road. For future work, we would like to simulate the protocol we propose on a network simulator and test its efficiency by simulating various infiltration attacks.

ACKNOWLEDGMENTS

This work is supported by CHANAKYA Fellowships of IITI DRISHTI CPS Foundation under the National Mission on Interdisciplinary Cyber Physical System (NM-ICPS) of Department of Science and Technology, Government of India.

DATA AVAILABILITY STATEMENT

The data will be made available to the reader by sending a request through email to the corresponding author of this manuscript.

ORCID

Shashank Gupta https://orcid.org/0000-0002-2124-9388

REFERENCES

1. Jiang Q, Zhang X, Zhang N, Tian Y, Ma X, Ma J. Three-factor authentication protocol using physical unclonable function for IoV. *Comput Commun*. 2021;173:45-55.
2. Umar M, Islam SH, Mahmood K, Ahmed S, Ghaffar Z, Saleem MA. Provable secure identity-based anonymous and privacy-preserving inter-vehicular authentication protocol for VANETS using PUF. *IEEE Trans Veh Technol*. 2021;70(11):12158-12167.
3. Babu PR, Reddy AG, Palaniswamy B, Das AK. EV-PUF: lightweight security protocol for dynamic charging system of electric vehicles using physical unclonable functions. *IEEE Trans Network Sci Eng*. 2022;9(5):3791-3807.
4. Zhang J, Qu G. Physical unclonable function-based key sharing via machine learning for IoT security. *IEEE Trans Ind Electron*. 2019;67(8):7025-7033.
5. Lee J, Kim G, Das AK, Park Y. Secure and efficient honey list-based authentication protocol for vehicular ad hoc networks. *IEEE Trans Network Sci Eng*. 2021;8(3):2412-2425.
6. Liu Y, Wang Y, Chang G. Efficient privacy-preserving dual authentication and key agreement scheme for secure V2V communications in an IoV paradigm. *IEEE Trans Intellig Transp Syst*. 2017;18(10):2740-2749.
7. Javaid U, Aman MN, Sikdar B. A scalable protocol for driving trust management in internet of vehicles with blockchain. *IEEE Internet Things J*. 2020;7(12):11815-11829.
8. Asim M, Guajardo J, Kumar SS, Tuyls P. Physical unclonable functions and their applications to vehicle system security. *VTC Spring 2009-IEEE 69th Vehicular Technology Conference*. IEEE; 2009:1-5.
9. Butun I, Österberg P, Song H. Security of the internet of things: vulnerabilities, attacks, and countermeasures. *IEEE Commun Surv Tutor*. 2019;22(1):616-644.
10. Ahmim I, Ghoualmi-Zine N, Ahmim A, Ahmim M. Security analysis on "Three-factor authentication protocol using physical unclonable function for IoV". *Int J Inform Secur*. 2022;21:1-8.
11. Chanda S, Luhach AK, Alnumay W, Sengupta I, Roy DS. A lightweight device-level public key infrastructure with dram based physical unclonable function (PUF) for secure cyber physical systems. *Comput Commun*. 2022;190:87-98.
12. Aman MN, Chua KC, Sikdar B. Mutual authentication in IoT systems using physical unclonable functions. *IEEE Internet Things J*. 2017;4(5):1327-1340.
13. Chatterjee U, Govindan V, Sadhukhan R, et al. Building PUF based authentication and key exchange protocol for IoT without explicit CRPs in verifier database. *IEEE Trans Depend Secure Comput*. 2018;16(3):424-437.
14. Frikken KB, Blanton M, Atallah MJ. Robust authentication using physically unclonable functions. *International Conference on Information Security*. Springer; 2009:262-277.
15. Harishma B, Patranabis S, Chatterjee U, Mukhopadhyay D. POSTER: authenticated key-exchange protocol for heterogeneous CPS. In: *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*; 2018 (pp. 849–851).
16. Feng W, Qin Y, Zhao S, Feng D. AAoT: lightweight attestation and authentication of low-resource things in IoT and CPS. *Comput Networks*. 2018;134:167-182.
17. Gope P, Sikdar B. An efficient data aggregation scheme for privacy-friendly dynamic pricing-based billing and demand-response management in smart grids. *IEEE Internet Things J*. 2018;5(4):3126-3135.
18. Xie L, Wang W, Shi X, Qin T. Lightweight mutual authentication among sensors in body area networks through physical unclonable functions. *2017 IEEE International Conference on Communications (ICC)*. IEEE; 2017:1-6.
19. Lim D, Lee JW, Gassend B, Suh GE, Van Dijk M, Devadas S. Extracting secret keys from integrated circuits. *IEEE Trans Very Large-Scale Integrat (VLSI) Syst*. 2005;13(10):1200-1205.
20. Liu H, Liu W, Lu Z, Tong Q, Liu Z. Methods for estimating the convergence of inter-chip min-entropy of SRAM PUFs. *IEEE Trans Circ Syst I: Reg Pap*. 2017;65(2):593-605.
21. Scheibel M, Stüble C, Wolf M. Design and implementation of an architecture for vehicular software protection. In: *Embedded Security in Cars Workshop (ESCAR'06)*; 2006.
22. Adelsbach A, Huber U, Sadeghi AR. Secure software delivery and installation in embedded systems. *Embedded Security in Cars*. Springer; 2006:27-49.
23. Wolf C. Zero-knowledge and multivariate quadratic equations. In: *Workshop on Coding and Cryptography*; 2004.
24. Bogdanov A, Eisenbarth T, Rupp A, Wolf C. Time-area optimized public-key engines: MQ-cryptosystems as replacement for elliptic curves? *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer; 2008:45-61.
25. Li H, Dán G, Nahrstedt K. Proactive key dissemination-based fast authentication for in-motion inductive EV charging. *2015 IEEE International Conference on Communications (ICC)*. IEEE; 2015:795-801.
26. Li H, Dán G, Nahrstedt K. Portunes+: privacy-preserving fast authentication for dynamic electric vehicle charging. *IEEE Trans Smart Grid*. 2016;8(5):2305-2313.
27. Rabieh K, Wei M. Efficient and privacy-aware authentication scheme for EVs pre-paid wireless charging services. *2017 IEEE International Conference on Communications (ICC)*. IEEE; 2017:1-6.
28. Gunukula S, Sherif AB, Pazos-Revilla M, Ausby B, Mahmoud M, Shen XS. Efficient scheme for secure and privacy-preserving electric vehicle dynamic charging system. *2017 IEEE international conference on communications (ICC)*. IEEE; 2017:1-6.
29. Pazos-Revilla M, Alsharif A, Gunukula S, Guo TN, Mahmoud M, Shen X. Secure and privacy-preserving physical-layer-assisted scheme for EV dynamic charging system. *IEEE Trans Veh Technol*. 2017;67(4):3304-3318.
30. Nabil M, Bima M, Alsharif A, et al. Priority-based and privacy-preserving electric vehicle dynamic charging system with divisible e-payment. *Smart Cities Cybersecurity and Privacy*. Elsevier; 2019:165-186.

31. Hamouid K, Adi K. Privacy-aware authentication scheme for electric vehicle in-motion wireless charging. *2020 International Symposium on Networks, Computers and Communications (ISNCC)*. IEEE; 2020:1-6.
32. Roman LF, Gondim PR. Authentication protocol in CTNs for a CWD-WPT charging system in a cloud environment. *Ad Hoc Networks*. 2020;97:102004.
33. Babu PR, Amin R, Reddy AG, Das AK, Susilo W, Park Y. Robust authentication protocol for dynamic charging system of electric vehicles. *IEEE Trans Veh Technol*. 2021;70(11):11338-11351.
34. Roche F, Brandenburg S. Should the urgency of visual-tactile takeover requests match the criticality of takeover situations? *IEEE Trans Intellig Veh*. 2019;5(2):306-313.
35. Wang Y, Luan HT, Su Z, Zhang N, Benslimane A. A secure and efficient wireless charging scheme for electric vehicles in vehicular energy networks. *IEEE Trans Veh Technol*. 2021;71(2):1491-1508.
36. Abouyoussef M, Ismail M. Blockchain-based privacy-preserving networking strategy for dynamic wireless charging of EVs. *IEEE Trans Network Serv Manage*. 2021;19:1203-1215.
37. Maiti A, Schaumont P. Improved ring oscillator PUF: an FPGA-friendly secure primitive. *J Cryptol*. 2011;24(2):375-397.
38. Gao M, Lai K, Qu G. A highly flexible ring oscillator PUF. In: *Proceedings of the 51st Annual Design Automation Conference*; 2014:1-6.
39. Chatterjee U, Chakraborty RS, Mukhopadhyay D. A PUF-based secure communication protocol for IoT. *ACM Trans Embed Comput Syst (TECS)*. 2017;16(3):1-25.
40. Zuckerkandl E, Pauling L. Evolutionary divergence and convergence in proteins. *Evolving Genes and Proteins*. Academic Press; 1965:97-166.
41. Baruah B, Dhal S. A two-factor authentication scheme against FDM attack in IFTTT based smart home system. *Comput Secur*. 2018;77:21-35.
42. Choi D, Seo SH, Oh YS, Kang Y. Two-factor fuzzy commitment for unmanned IoT devices security. *IEEE Internet Things J*. 2018;6(1):335-348.
43. Ying B, Nayak A. Anonymous and lightweight authentication for secure vehicular networks. *IEEE Trans Veh Technol*. 2017;66(12):10626-10636.
44. Mohit P, Amin R, Biswas GP. Design of authentication protocol for wireless sensor network-based smart vehicular system. *Veh Commun*. 2017;9:64-71.
45. Ma M, He D, Wang H, Kumar N, Choo KKR. An efficient and provably secure authenticated key agreement protocol for fog-based vehicular ad-hoc networks. *IEEE Internet Things J*. 2019;6(5):8065-8075.
46. Li X, Liu T, Obaidat MS, Wu F, Vijayakumar P, Kumar N. A lightweight privacy-preserving authentication protocol for VANETs. *IEEE Syst J*. 2020;14(3):3547-3557.
47. Gazdar T, Rachedi A, Benslimane A, Belghith A. A distributed advanced analytical trust model for VANETs. *2012 IEEE Global Communications Conference (GLOBECOM)*. IEEE; 2012:201-206.
48. Haddadou N, Rachedi A, Ghamri-Doudane Y. A job market signaling scheme for incentive and trust management in vehicular ad hoc networks. *IEEE Trans Veh Technol*. 2014;64(8):3657-3674.
49. Yahiatene Y, Rachedi A. Towards a blockchain and software-defined vehicular networks approaches to secure vehicular social network. *2018 IEEE Conference on Standards for Communications and Networking (CSCN)*. IEEE; 2018:1-7.
50. Lu Z, Wang Q, Qu G, Liu Z. BARS: a blockchain-based anonymous reputation system for trust management in VANETs. *2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE)*. IEEE; 2018:98-103.
51. Lu Z, Liu W, Wang Q, Qu G, Liu Z. A privacy-preserving trust model based on blockchain for VANETs. *IEEE Access*. 2018;6:45655-45664.
52. Yang Z, Yang K, Lei L, Zheng K, Leung VC. Blockchain-based decentralized trust management in vehicular networks. *IEEE Internet Things J*. 2018;6(2):1495-1505.
53. Malik N, Nanda P, Arora A, He X, Puthal D. Blockchain based secured identity authentication and expeditious revocation framework for vehicular networks. *2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE)*. IEEE; 2018:674-679.
54. Lei A, Cruickshank H, Cao Y, Asuquo P, Ogah CPA, Sun Z. Blockchain-based dynamic key management for heterogeneous intelligent transportation systems. *IEEE Internet Things J*. 2017;4(6):1832-1843.
55. Khelifi H, Luo S, Nour B, Moungla H, Ahmed SH. Reputation-based blockchain for secure NDN caching in vehicular networks. *2018 IEEE Conference on Standards for Communications and Networking (CSCN)*. IEEE; 2018:1-6.
56. Singh M, Kim S. Trust bit: reward-based intelligent vehicle commination using blockchain paper. *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*. IEEE; 2018:62-67.
57. Zhang X, Chen X. Data security sharing and storage based on a consortium blockchain in a vehicular ad-hoc network. *IEEE Access*. 2019;7:58241-58254.
58. Zhang X, Li R, Cui B. A security architecture of VANET based on blockchain and mobile edge computing. *2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN)*. IEEE; 2018:258-259.
59. Shrestha R, Bajracharya R, Nam SY. Blockchain-based message dissemination in VANET. *2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS)*. IEEE; 2018:161-166.
60. Singh M, Kim S. Crypto trust point (cTp) for secure data sharing among intelligent vehicles. *2018 International Conference on Electronics, Information, and Communication (ICEIC)*. IEEE; 2018:1-4.
61. Islam SH, Obaidat MS, Vijayakumar P, Abdulhay E, Li F, Reddy MKC. A robust and efficient password-based conditional privacy preserving authentication and group-key agreement protocol for VANETs. *Fut Gen Comput Syst*. 2018;84:216-227.
62. Cui J, Tao X, Zhang J, Xu Y, Zhong H. HCPA-GKA: a hash function-based conditional privacy-preserving authentication and group-key agreement scheme for VANETs. *Veh Commun*. 2018;14:15-25.

How to cite this article: Katyal S, Gupta S, Rawlley O, Ghosh D. A fog-driven three-factor authentication protocol for secure data sharing in Internet of Vehicles cyber-physical systems. *Concurrency Computat Pract Exper*. 2024;36(8):e7981. doi: 10.1002/cpe.7981