Towards Vehicular Digital Forensics from Decentralized Trust: An Accountable, Privacy-preserving, and Secure Realization

Ming Li, Jian Weng*, *Member, IEEE*, Jia-Nan Liu, *Student Member, IEEE*, Xiaodong Lin, *Fellow, IEEE*, Charlie Obimbo,

***Abstract*—With the increasing number of traffic accidents and terrorist attacks by modern vehicles, vehicular digital forensics (VDF) has gained significant attention in identifying evidence from the related digital devices. Ensuring the law enforcement agency to accurately integrate various kinds of data is a crucial point to determine the facts. However, malicious attackers or semi-honest participants may undermine the digital forensic procedures. Enabling accountability and privacy-preservation while providing secure data access control in VDF is a non-trivial challenge. To mitigate this issue, in this paper, we propose a blockchain-based decentralized solution for VDF named BB-VDF, in which the accountable protocols and privacy-preserving algorithm are constructed. The desirable security properties and fine-grained data access control are achieved based on smart contract and the customized cryptographic construction. Specifically, we design a distributed key-policy attribute based encryption scheme with partially hidden access structures, named DKP-ABE-H, to realize the secure fine-grained forensics data access control. Further, a novel smart contract is designed to model the forensics procedures as a finite state machine, which guarantees accountability that each participant performs auditable cooperation under tamper-resistant and traceable transactions. Systematic security analysis and extensive experimental results show the feasibility and practicability of our proposed BB-VDF scheme.**

***Index Terms*—Blockchain, accountability, privacy-preservation, vehicular digital forensics.**

I. Introduction

THe functionalities of vehicles have been strengthened tremendously with the increasing in-vehicle sensors, control units, and communication methods, such as Electronic Control Unit (ECU), Bluetooth, and Wi-Fi. According to statistics from *Ford Motor Company* [1], a modern-day vehicle has approximate 50-70 computers, which enables it to be an important source of digital data (e.g., locations where doors are open/close, when the tires are pumped and lights are on/off). These wealth of sensing and operation data make vehicles more intelligent and smart, which will prompt the prosperity of the autonomous driving industry in the near future [2].

M. Li, J. Weng, and J. Liu are with the College of Cyber Security, Jinan University, Guangzhou 510632, China. Jian Weng is the corresponding author. E-mail: cryptjweng@gmail.com, limjnu@gmail.com.

X. Lin and C. Obimbo are with the School of Computer Science, University of Guelph, Guelph ON N1G 2W1, Canada. E-mail: xlin08@uoguelph.ca, cobimbo@uoguelph.ca.

However, as everything has its two sides, the increasing number of vehicles also brings us lots of security issues. Specifically, using vehicles as weapons to conduct terrorist attacks are not rare and have caused tremendous damages to our society [3], [4]. Vehicle Ramming Attack (VRA), one of the typical attacks in reality, refers to malicious attackers deliberately using a vehicle to ram a building or a crowd of people. For example, on July 14, 2016, a 20-ton rental truck rammed into the crowd who were watching a firework display in Nice, France, which resulted in the deaths of 86 people and wounded more than 450 people [5]. Moreover, according to a *cnn.com* report on terrorist attacks caused by vehicles[1], at least 7 major attacks happened in 2017, which led to at least 37 people dead and hundreds of pedestrians injured. Since it is incredibly easy to get one car from rental companies, many terrorist attackers choose a rental car as the criminal weapon[2]. In other words, launching a VRA requires minimal capability but can cause catastrophic disasters to society (*vehicle* and *car* are used interchangeably in this paper).

To mitigate these types of VRAs, forensics investigation specialized for vehicles can be conducted to analyze the suspicious behaviors and collect evidences, which is called *Vehicular Digital Forensics* [6] (VDF) (also called "Vehicle Forensics" [7]). VDF has gained considerable attention both in academic and industrial area [8] since a vast amount of data is being collected by in-vehicle computers. It can help law enforcement agencies to detect a potential VRA by identifying suspicious activities. In particular, this field becomes more significant with the forthcoming of car-sharing and self-driving cars, which are the way of the future [2]. However, it also brings a burning question: *who is at fault if a self-driving car gets involved in a fatal accident, the driver or the car manufacturer who develops the self-driving algorithms?* If it is in the latter case, the manufacturer could be sued for an unprecedented amount of money for a lost life, and eventually goes out of business. Thus, it has become crucial to have a forensically sound way for authorities to investigate car accidents in the era of autonomous driving.

As for VRA, the law enforcement agency may prevent it from happening beforehand if he obtains valuable data by VDF. For instance, the rental company can detect that someone

[1]https://www.bbc.com/news/world-europe-37137816

[2]https://www.cnn.com/2017/05/03/world/terrorist-attacks-by-vehicle-fast-facts/index.html

has difficulty in explaining the purposes of renting a car. Combining with other related data, such as traffic management center to report that the car parks in a specific area for several days without any reasonable explanation, the law enforcement agency (i.e., the investigator) may conjecture that it is a potential threat to the public safety in this area [3]. Obviously, a single data source is not enough for the analysis of suspicious behaviors, comprehensive historical data from the car and related data sources should be obtained by the investigator.

Unfortunately, there exist several security and privacy issues that may have adverse impacts on the implementation of VDF: (i) Firstly, the details of the warrant may be leaked to potential criminals by malicious insiders, which allows criminals to alert and changes their behavior temporarily. (ii) Secondly, the investigator may violate the purview of the warrant to require more data than being authorized. He may even tamper with the collected evidence. (iii) Thirdly, there exist malicious insiders who might modify the historical evidence before presenting to the investigator, or claim the historical evidence has been lost, which apparently violates the digital chain of custody [9]. Apart from these issues, other problems which are adverse to the normative VDF procedures also exist. In particular, as vehicles become more complicated than before, it is hard for the law enforcement agency to obtain valid forensics data due to the lack of specialized skills, but to resort to a commercial party for help. However, it thus brings a potential threat that the privacy of the investigation may be exposed. Besides, since the court has released a large number of warrants accumulatively, it is easy to forget to trace the state of warrants, allowing semi-honest investigators to still use expired warrants to request data [10].

We note that several efforts have been made to address parts of the issues [10]–[14], while most of them focus on different aspects of digital forensics, and have different system models and security models. In particular, we consider that the public should be able to audit the process of VDF without privacy leakage, which can assure the legitimacy of forensics process without powers abusing or misusing. Besides, the privacy is a fundamental human right that data should be collected with fine-grained access control during the investigation, nothing more and nothing less. It is non-trivial to resolve the above security and privacy issues in the VDF simultaneously. The closest to our work is [11], which is claimed as the first research to propose a blockchain-based framework to conduct vehicle forensics by integrating different kinds of data. However, they do not resolve the challenges that (i) the confidentiality of warrants should be preserved, and (ii) the involved parties (e.g., the law enforcement agency) might behave dishonestly.

Towards this end, we explore the potential of blockchain, and take advantage of cryptographic primitives to design a blockchain-based solution for VDF with achieving accountability, privacy preservation, and fine-grained data access control. We construct a permissioned blockchain to integrate data from different data sources. The procedures of VDF is modeled as a finite-state machine (FSM) in smart contracts, which enables involved parties to cooperate legally under the supervision of the public. Our **contributions** can be summarized as follows:

- We formalize a distributed KP-ABE with partially hidden access structures scheme (short for DKP-ABE-H) to realize fine-grained access control while preserving privacy of access structures, i.e., the attributes. We prove the security of DKP-ABE-H under the typical selective-set model and show its usability in the digital forensics scenario.
- We propose a blockchain-based solution for the vehicular digital forensics called BB-VDF with achieving accountability and privacy preservation. The BB-VDF scheme guarantees the privacy of the warrant details and forensics data by leveraging the proposed DKP-ABE-H scheme, and also eliminates the issues of single point of failure/compromise (SPoF/C) based on blockchain. Further, we design a set of customized smart contracts that model the vehicle forensics process as an FSM in the blockchain. Such FSM follows the *verification-then-forwarding* pattern that enforces the involved parties to cooperate honestly in the VDF.
- We implement a prototype of the proposed scheme and deploy it to the Ethereum public test network *Rinkeby*. Extensive experimental results indicate the feasibility and practicability of the proposed BB-VDF scheme.

Paper Structure. The remainder of the paper is organized as follows. In the next Section, we introduce the background and related preliminaries. In Section III, we formalize the system model and threat model. In Section IV, we present the construction of DKP-ABE-H and analyze its security. In Section V, we present the proposed concrete scheme, including security proof and analysis. In Section VI, we present the experiments and evaluation results. The related works are given in Section VII. Finally, we give the conclusion in Section VIII.

II. Background and Preliminaries

A. Digital Forensics

As illustrated in Fig. 1, a typical digital forensics can be depicted as two main phases: (i) the first phase is about warrant authorization that law enforcement agency (e.g., the policeman) requests an authorization from the court before accessing data of any individual entity [15]. The warrant contains a signature from the court to permit the law enforcement agency to conduct an investigation. (ii) The second phase is on data processing that contains four steps: *collection, examination, analysis*, and *reporting* [16]. More concretely, *collection* is the process of evidence collection that aims to gather sufficient data from the software system (e.g., mobile App) or hardware devices (e.g., physical RAM and SD card). These data and devices should be stored with due care to protect the integrity and confidentiality. *Examination* is to perform a search of the data that is related with the respondent or the suspected crime. *Analysis* aims to conduct more systematical and professional analysis on the collected data or devices. *Reporting* is responsible for providing the final investigation reports on the results of the previous process.

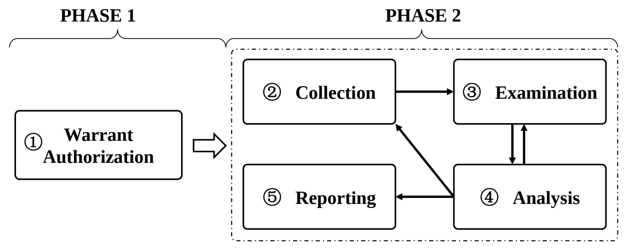


Fig. 1. The workflow of digital forensics.

B. Blockchain and Smart Contract

Most recently, the blockchain technology has been employed in various applications, such as financial services, healthcare [17], internet-of-things [18], [19] and crowdsourcing [20], [21]. It is essentially a distributed ledger that is maintained by a number of network nodes (also called blockchain nodes) [22], [23]. These nodes might be mutually distrustful while can still reach an agreement based on the consensus protocol, e.g., proof of work (PoW) or proof of stake (PoS). More precisely, the blockchain is composed of a series of consecutive *blocks*, i.e, an ordered hash chain, where each block can be created by a blockchain node through a proof, e.g., a specific hash value that can satisfy the difficulty level. A block contains a number of *transactions* which are the core parts of the blockchain. One can complete a coin payment through a transaction which specifies the sender address, receiver address, the amount and timestamp. In addition, there exists a data field that can be used to realize script-based payment logic, e.g., the time-lock deposit. The security of blockchain is guaranteed by the underlying cryptographic primitives that ensure the transmissions of digital currency or status transitions among different entities in a secure way.

Particularly, the review of main features on blockchain can be listed as follows: (i) *Complete Decentralization*: it is based on a distributed P2P network that many untrusted nodes can achieve fair data exchange without reliance on a central party. (ii) *Correct Execution*: blockchain is a global computer that each blockchain node can trace and verify the correctness of the data computation. (iii) *Tamper-resistance*: the data (i.e., blocks and transactions) are tamper-resistant since they are organized in a special data structure (Merkle tree and hash chain).

And also, smart contracts are designed to construct the decentralized application (DApp) [24] that facilitates the process of an application to be executed automatically and verifiably. One can participate in a DApp by providing valid inputs through on-chain transactions to call a function of smart contract.

C. Cryptography Algorithms

In our constructions, we make use of the following cryptographic algorithms as building blocks to achieve accountability and fine-grained access control.

Bilinear Pairing: Let $\mathbb{G}_1$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p. g is a random generator of $\mathbb{G}_1$. Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ be a computable bilinear pairing with the follow properties:

- Bilinearity: for all $g \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_p$, we have $e(g^a, g^b) = e(g^b, g^a) = e(g, g)^{ab}$.
- Non-degeneracy: $e(g, g) \neq 1$.
- Computability: the bilinear mapping and the group computation in $\mathbb{G}_T$ are efficiently computable.

Distributed Key Generation (DKG): DKG is one of the components in (t, n)-threshold cryptosystem [25]. It allows n parties to collectively generate a key pair (i.e., public key and private key) without letting any single party reconstruct the secret key. Besides, it also does not rely on any trusted party to achieve t-secure, which means that the protocol is secure if no more than $t + 1$ parties are broken. Further, Gennaro *et al.* [26] improved the security of DKG protocol with the uniform randomness property. In [26], each honest party holds a share α_i of a secret key α. For any each set $\mathcal{N}$ of $t + 1$ correct shares, $\alpha = \sum_{i \in \mathcal{N}} \lambda_i \cdot \alpha_i$, where λ_i are Lagrange interpolation coefficients for set $\mathcal{N}$. Specially, the t-secure DKG protocol will always satisfy the following *correctness* and *secrecy* properties:

- *Correctness:* Any subsets of $t+1$ shares define the same privacy key α ($\alpha \in \mathbb{Z}_p$) and all parties share the same public key $y = g^{\alpha}$.
- *Secrecy:* There is no information learned on α expect for the implication of value $y = g^{\alpha}$.

Key Policy Attribute-Based Encryption (KP-ABE): KP-ABE scheme is a type of public key encryption. It allows users to encrypt and decrypt data based on attributions. Compared with the identity-based encryption (IBE) scheme, KP-ABE scheme is more suitable to support fine-grained access control policy. Generally, KP-ABE consists of the following four algorithms [27]:

-*Setup*$(1^{\eta}) \rightarrow (PK, MSK)$. *The setup algorithm takes a security parameter* η *as the input and outputs the public parameters* PK *and a master secret key* MSK.

-*Encrypt*$(PK, M, S) \rightarrow CT$. *The encryption algorithm takes the public parameters* PK*, a set of attributes* S *and a message* M *as the inputs. It selects a random number* $s \in \mathbb{Z}_p$ *outputs a ciphertext* CT.

-*KeyGen*$(PK, MSK, \mathbb{A}) \rightarrow SK$. *The key generation algorithm takes the public parameters* PK*, the master secret key* MSK *and an access structure* $\mathbb{A}$ *as the inputs and outputs the private key* SK.

-*Decrypt*$(PK, SK, CT) \rightarrow M$. *The decryption algorithm takes public parameters* PK*, a private key* SK*, and a ciphertext* CT *associated with a set of attributes* S *as inputs and returns a message* M.

III. System and Security Models

In this section, we formally present the blockchain-based VDF framework and system model. Then, we give our security assumptions and discuss the threat model. Finally, the security goals are clearly defined. The notations that will be utilized are presented in TABLE I.

TABLE I
THE NOTATIONS OF EXPLANATION.

Notation	Explanation
$\mathcal{L}$	The law enforcement agency.
$\mathcal{C}$	The court.
$\mathcal{B}$	The blockchain platform.
$\mathcal{D}$	The distributed data storage system.
$[N], [\ell]$	The sequence number of $(1, \ldots, n)$, $(1, \ldots, \ell)$, respectively.
$(\mathcal{A}_1, \ldots, \mathcal{A}_n)$	The n decryption authorities.
$(\mathcal{S}_1, \ldots, \mathcal{S}_m)$	The data sources.
PK, MSK	The public parameters and master secret key.
$id, type, t$	The identifier, type and timestamp of forensics data.
K_e^p, K_e^s	The party's public key and private key.
$M_1 \| M_2$	The concatenation of message M_1 and M_2.
H_0, H_1, H_2	Three non-cryptographic hash functions.
$\mathsf{Enc}_{sk}(M)$	The symmetric encryption on message M with private symmetric key sk, e.g., AES.
$r', r_x, r_{x,y}$	The generated random numbers.
$\mathbb{A} = (W, \rho)$	The access structure in KP-ABE.
Ti	The timestamp in the transaction.

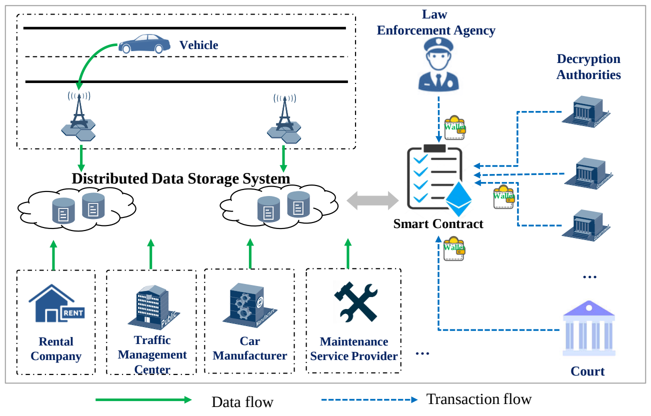


Fig. 2. System model.

A. System Architecture

The architecture of the proposed scheme is shown in Fig.2, which consists of six entities: Data Sources, Law Enforcement Agency, Court, Decryption Authorities, Blockchain Nodes, Data Storage Nodes. Each entity has an initial key pair (i.e., public key and private key).

- Data Sources: identified by $\mathcal{S} = \{\mathcal{S}_1, \ldots, \mathcal{S}_m\}$, refer to the parties who generate the data which are helpful for VDF and stores them to the data storage based on forensics-by-design paradigm [28].
- Law Enforcement Agency: identified by $\mathcal{L}$, refers to the investigator (e.g., the policeman) who is responsible for launching a digital forensics investigation. $\mathcal{L}$ is assumed to have professional skills (including software and hardware skills) to collect valuable data from different data sources.
- Court: identified by $\mathcal{C}$, refers to the official judges who are responsible for approving $\mathcal{L}$'s request to conduct an investigation by following specified legal standards.
- Decryption Authorities, identified by $\mathcal{A} = \{\mathcal{A}_1, \ldots, \mathcal{A}_n\}$, refer to the parties who jointly maintain a master secret key. They provide the shares to allow $\mathcal{L}$ to recover the decryption key if he has an authorized warrant. $\mathcal{A}$ can be the established organizations in real-world deployment, e.g., the government departments.
- Blockchain Nodes: recognized as a permissioned blockchain (identified by $\mathcal{B}$) that is maintained by multiple members, such as the court, the law enforcement agency. Other parties are allowed to join in the blockchain with the permission.
- Data Storage Nodes, refer to data nodes who store the related forensics data. Our scheme adopts the distributed data storage techniques, e.g., S3. Data is encrypted in $\mathcal{D}$ so that if $\mathcal{L}$ intends to obtain data, he should be granted with a corresponding decryption key.

At the beginning, the authorities initialize the system by generating the public parameters. Then, upon receiving a report on a suspicious behavior or a traffic accident, $\mathcal{L}$ requests a warrant from $\mathcal{C}$. If $\mathcal{C}$ permits, he will issue an authorized warrant which is cryptographically signed by her/his secret key to $\mathcal{L}$. Then, $\mathcal{L}$ can require a decryption key from $\mathcal{A}$ using the warrant and collect data from $\mathcal{D}$. During the investigation, each procedure is completed with a transaction which prompts the state machine transition in smart contracts and enables the public to audit the validity and legitimacy of the investigation. The underlying $\mathcal{B}$ can be constructed atop an existing blockchain, e.g., Enterprise Ethereum Blockchain. It supports Turing-complete smart contracts (e.g., *solidity*), which is extremely useful for constructing the state machine and accomplishing auditable forensics investigations. In practice, the involved entities can be motivated by rewards and penalties in smart contracts[3].

Particularly, to collect data from vehicles, a *forensics daemon* is assumed to run inside OBU to communicate with roadside units, and submit the real-time data to $\mathcal{B}$ and $\mathcal{D}$ periodically. In addition, an embedded hardware-security-module (HSM) based scheme [29] is adopted that data requests should be authorized previously, ensuring ECUs with secure communications for on-board systems. To guarantee the security of data collection, the forensics daemon will communicate with different ECUs through Controller Area Network (CAN) bus and require access authorization.

B. Security Model

Here, we discuss security threats on digital forensics. Without loss of generality, the security of $\mathcal{B}$ follows the *majority-honest-assumption* [20], [30], that is, if most of the blockchain nodes are honest, $\mathcal{B}$ will perform the predefined computation correctly. The forensics daemon is assumed to run securely inside OBU so that malicious attackers can not disturb its normal execution, or tamper the content of collected data. Similar to the previous schemes on digital forensics [16], [31], [32], most of the entities are honest and perform their duties properly with bounded computation. Moreover, $\mathcal{L}$, $\mathcal{C}$, and honest $\mathcal{A}$ do not collude with each other during the investigation. However, there might exist some malicious insiders or external attackers who may undermine the forensics investigation. In particular, we focus on security threats as follows:

[3]We leave the design of incentive mechanisms as our future work

Threat 1: Untrustworthy Law Enforcement Agency $\mathcal{L}$. Generally, $\mathcal{L}$ belongs to a trusted institution in reality. However, there exist some untrustworthy insiders who are in pursuit of individual profit and do not follow the standardized digital forensics process. More precisely, a warrant authorized by $\mathcal{C}$ has designated the data range, which specifies that $\mathcal{L}$ can not acquire data more than the designated during the investigation. However, untrustworthy $\mathcal{L}$ (denoted as $\mathcal{L}^*$) may attempt to (i) acquire more data without explicit authorization, (ii) take advantage of an expired warrant to obtain access to data, or (iii) alter or forge the collected evidences before presenting to $\mathcal{C}$. Furthermore, potential external attackers may attempt to compromise $\mathcal{L}$ and impersonate an authorized $\mathcal{L}$ to conduct further attacks.

Threat 2: Honest-but-Curious Court. Similarly, $\mathcal{C}$ is a trusted institution who complies with the designated protocols to make a judgement. $\mathcal{C}$ can not learn about the details of data records besides the related forensics data. However, there may exist malicious insiders who are curious about the detailed forensics data which are not applied by $\mathcal{L}$. In addition, $\mathcal{C}$ might forget to track the state of an authorized warrant, which allows $\mathcal{L}$ to acquire data by the expired warrants.

Threat 3: Untrustworthy Authorities. None of $\mathcal{A}$ can learn about the details of warrant and forensics data. Given an authorized warrant, $\mathcal{A}$ will provide the correct secret shares honestly. However, partial compromised or internal untrustworthy $\mathcal{A}$ (denoted as $\mathcal{A}^*$, no more than a predefined ratio) might exist. They attempt to learn and expose the details of an investigation, e.g., the identity of a suspicious vehicle. In addition, they may collude with suspicious vehicles, allowing them to change their behaviors.

C. Security Goals

To resist the aforementioned security threats, and realize accountability and fine-grained access control for VDF, the proposed scheme will achieve the following high-level security goals:

Accountability Accountability in digital forensics can be considered as secure assurance that the related entities will not misuse or abuse their powers during the investigation forensics. Our proposed scheme achieves accountability from the following two aspects:

- *Complete VDF Audit:* To ensure the legitimacy of an investigation, the crucial procedures should be identified and each procedure can be audited by the public under the blockchain-based model.
- *Public Verifiability:* The proposed scheme should be able to prevent unauthorized behaviors. Specifically, each procedure should be publicly scrutinized that the involved parties are accountable for the investigation and prevented from abusing or misusing their granted power.

Privacy Preservation The proposed scheme should preserve the privacy of warrant and forensics data as follows:

- *Confidentiality of Warrant:* The detailed information of a warrant should be temporarily protected to prevent unauthorized entities from learning about it for a period of time. We design that $\mathcal{L}$ and $\mathcal{C}$ can know the details of warrant, while other unauthorized parties can only know the digest of the warrant.
- *Confidentiality of Forensics Data:* The plaintext of forensics data should be protected except the authorized agency $\mathcal{L}$.

IV. The Proposed DKP-ABE-H

In this section, we present the construction of DKP-ABE-H which is a critical building block for the BB-VDF scheme.

A. Syntax of DKP-ABE-H

Formally, the proposed decentralized key-policy attribute-based encryption with partially hidden access structures scheme DKP-ABE-H=(Setup, Encrypt, SubPolicyHiddenGen, SubKeyGen, KeyAggre, Decrypt) is defined by six polynomial-time computable algorithms. Each algorithm is run by one of the three entities: the sender, the receiver, or the authorities:

- Setup$(1^\eta, t, n)$*: is a stateful setup algorithm run by the authorities* $\mathbb{A}$ *collaboratively. It chooses the security parameter* $\eta \in \mathbb{Z}_p$*, key management parameters* t *and* n*, and key pairs owned by* $\mathbb{A}$ *as inputs, and outputs the master public key* PK*. The master private key* MSK *is privacy-preserved so that no one could recover it.*
- Encrypt(M, PK, S) *is a probabilistic encryption algorithm run by the sender. It takes a message* M*, the attributes* S *and* PK *as inputs, and outputs the ciphertext* C*.*
- SubPolicyHiddenGen$(\mathbb{A})$ *is a probabilistic encryption algorithm run by the receiver. It outputs the intermediate parameters* $\{\varphi_j\}_{j\in[\ell]}$ *which are used to recover the decryption key. These parameters can be public so that others can not learn the attributes.*
- SubKeyGen$(\{\varphi_j\}_{j\in[\ell]}, \alpha_i)$ *is a probabilistic encryption algorithm run by an authority* $\mathcal{A}_i$*. It takes partial policies* $\{\varphi_j\}_{j\in[\ell]}$ *and* $\mathcal{A}_i$*'s secret key* α_i *of as inputs, and outputs a share of the decryption key* $\widetilde{SK_i}$*.*
- KeyAggre$(\{\widetilde{SK_\tau}\}_{\tau\in\mathcal{N}})$ *is a deterministic key aggregation algorithm run by the receiver. It takes any* $\mathcal{N}$ *authorities' shares as inputs and outputs the decryption key* SK*.*
- Decrypt(SK, C) *is a deterministic key aggregation algorithm run by the receiver. It takes* SK *and* C *as inputs and outputs the message* M*.*

B. Security Definition of DKP-ABE-H

Our security model of DKP-ABE-H is similar with the typical selective-set model as in [33], [34]. Specifically, we consider the following experiment $Exp_{\mathcal{AT},\Pi}(\eta, S)$ for an adversary $\mathcal{AT}$ and a challenger $\mathcal{C}$:

Init. The adversary $\mathcal{AT}$ chooses a challenge attribute set S^* and sends it to the challenger.

Setup. The challenger $\mathcal{C}$ runs the Setup algorithm to generate the system parameters, and gives the public parameters PK to the adversary while keeping the master secret key MSK.

Phase 1. The challenger $\mathcal{C}$ initializes an empty table T, two empty sets D, E and two integer counters $j = 0$, $l = 0$. The adversary $\mathcal{AT}$ can adaptively and repeatedly issue the following two queries:

- Create($\mathbb{A}$): If S^* satisfies $\mathbb{A}$, then it returns $\perp$. The challenger sets $j = j + 1$. It executes the $\mathsf{SubPolicyHiddenGen}(\mathbb{A})$, $\mathsf{SubKeyGen}(\{\varphi_j\}_{j\in[\ell]}, \alpha_i)$ and $\mathsf{KeyAggre}(\{\widetilde{SK_\tau}\}_{\tau\in\mathcal{N}})$ algorithms to obtain the private key SK and stores the entry $(i, \mathbb{A}, SK)$ into the table T.
- CorruptSK(i): If there exists an i-th entry in table T, the challenger obtains the entry $(i, \mathbb{A}, SK)$, and sets $D = D\cup\{\mathbb{A}\}$. Then it sends the private key SK to the adversary. Otherwise, it returns $\perp$.
- CorruptPartialMSK(k): If $l \geq t$, it returns $\perp$. The challenger sets $l = l + 1, E = E \cup \{\alpha_k\}$ and returns the partial MSK α_k to the adversary.

Challenge. The adversary chooses two messages M_0^* and M_1^* as challenge messages. The challenger randomly chooses a bit $b \in \{0, 1\}$ and encrypts M_b by running $\mathsf{Encrypt}(M_b^*, PK, S^*)$ algorithm. The output C^* is given to the adversary.

Phase 2. Phase 1 is repeated with the following restrictions:

- The adversary cannot obtain the secret key for the challenge ciphertext C^*. That is, the adversary cannot issue a CorruptSK query to obtain a secret key SK that the corresponding access structure $\mathbb{A}$ which S^* satisfies.
- The adversary cannot issue the CorruptPartialMSK query for more than t times.

Guess. The adversary outputs the guess b' of b. The experiment outputs 1 iff $b = b'$.

Definition 2. *A decentralized key-policy attribute-based encryption (DKP-ABE-H) is CPA-secure in the selective model, if for all probabilistic polynomial-time adversary, the advantage of the adversary in the $Exp_{\mathcal{AT},\Pi}(\eta, S^*)$ satisfied:*

$$Pr[Exp_{\mathcal{AT},\Pi}(\eta, S^*) = 1] - 1/2 \leq negl(\eta), \tag{1}$$

where η is the security parameter.

C. Construction of DKP-ABE-H

$\mathsf{Setup}(1^\eta, t, n) \rightarrow \{PK, \{\alpha_i\}_{i\in[N]}\}$ is a stateful setup algorithm. Given the security parameters η, it generates the public parameters. Let $\mathbb{G}_1$ be a bilinear group with prime order $p \in \Theta(2^\eta)$, g is a random generator of $\mathbb{G}_1$. H_0 is a cryptographic hash function: $H_0 : \{0, 1\}^* \rightarrow \mathbb{G}_1$. Specifically, each authority $\mathcal{A}_j$ holds a secret share α_j and jointly maintains a master secret key α with other authorities based on the secure DKG protocol [26]. After that, a set $\mathcal{N}$ of $t + 1$ authorities publish their partial public parameters $v_j = e(g, g)^{\alpha_j}$ and generate the system public parameter:

$$\prod_{j\in\mathcal{N}} \left(e(g, g)^{\alpha_j}\right)^{\lambda_j} = e(g, g)^{\sum\limits_{j\in\mathcal{N}} \lambda_j\cdot\alpha_j} = e(g, g)^\alpha, \tag{2}$$

where λ_j is the Lagrange interpolation coefficient for sharing secret α. The authorities collaboratively specify the master public parameters $PK = (\mathbb{G}, p, g, e(g, g)^\alpha, H_0)$. Note that the master secret key $MSK = \{\alpha\}$ is not controlled by any party.

$\mathsf{Encrypt}(M, PK, S) \rightarrow C$ is a probabilistic encryption algorithm. It randomly generates one number $s \in \mathbb{G}_p$, and takes the master public parameters PK, a message $M \in \mathbb{G}_T$ and an attribute set S as inputs. The output ciphertext is computed as follows:

$$C = M\cdot e(g, g)^{\alpha s}, \hat{C} = g^s, \{C_x = H_0(x)^s\}_{x\in\{0,1\}^*}. \tag{3}$$

$\mathsf{SubPolicyHiddenGen}(\mathbb{A}) \rightarrow \{\varphi_j\}_{j\in[\ell]}$ is a probabilistic partial policy generation algorithm, where $\mathbb{A} = (W, \rho)$, ρ is a function that associates rows of W (an $\ell \times n$ matrix) to the attributes. To preserve the privacy of access structure, the attribution values $\rho(1), ..., \rho(\ell)$ are not revealed to any third party. To achieve this, the receiver chooses a set of random numbers $\{r_i, r_{i,d}\}_{i\in[\ell]} \in \mathbb{Z}_p$ and computes blinded values as follows:

$$\begin{aligned}
&\varphi_1 = (H_0(\rho(1))^{r_1}, g^{r_1 r_{1,1}}, \forall d \in \Gamma/\rho(1), H_0(d)^{r_1 r_{1,d}}),\\
&...\\
&\varphi_\ell = (H_0(\rho(\ell))^{r_\ell}, g^{r_\ell r_{\ell,1}}, \forall d \in \Gamma/\rho(\ell), H_0(d)^{r_\ell r_{\ell,d}}),
\end{aligned} \tag{4}$$

where $\Gamma = \{d : \exists i \in [\ell], \rho(i) = d\}$. $\{\varphi_1, ..., \varphi_\ell\}$ are sent to the authorities. Note that $\varphi_i = \{H_0(\rho(i))^{r_i}, g^{r_i r_{i,1}}, \forall d \in \Gamma/\rho(i), H_1(\rho(d))^{r_i r_{i,d}}\}, i \in [\ell]$ have the random number $r_{i,1}, r_{i,d}$, such that the authorities cannot learn the values of the attributes by launching guessing attack, e.g., using the pairing algorithm.

$\mathsf{SubKeyGen}(\{\varphi_j\}_{j\in[\ell]}, \alpha_i) \rightarrow \widetilde{SK_i}$ is a probabilistic partial secret key generation algorithm. Specifically, an honest authority $\mathcal{A}_j$ utilizes the share of the master secret α_j to generate a share of the decryption key. Since the master secret key α are not kept by any single authority, $\mathcal{A}_j$ cannot compute $\{g^{\mu_1}, ..., g^{\mu_\ell}\}$ directly as in [27], while he can use α_j to compute the intermediate parameters $\{g^{\mu_{1,j}}, ..., g^{\mu_{\ell,j}}\}$ as follows:

$$g^{\mu_{i,j}} = g^{W_i\cdot\vec{v_j}} = g^{W_i\cdot(\alpha_j, y_2, ..., y_n)}, i \in [\ell]. \tag{5}$$

Specially, to prevent the receiver from learning the value $g^{\mu_{i,j}}$, $\mathcal{A}_j$ chooses a random number $r'_j \in \mathbb{Z}_p$, and computes the decryption key share $\widetilde{SK_j} = (\widetilde{\varphi_{1,j}}, ..., \widetilde{\varphi_{\ell,j}})$ as follows:

$$\begin{aligned}
&\widetilde{\varphi_{1,j}} = (g^{\mu_{1,j}} H_0(\rho(1))^{r_1 r'_j}, g^{r_1 r_{1,1} r'_j}, \forall d \in \Gamma/\rho(1), H_0(d)^{r_1 r_{1,d} r'_j}),\\
&...\\
&\widetilde{\varphi_{\ell,j}} = (g^{\mu_{\ell,j}} H_0(\rho(\ell))^{r_\ell r'_j}, g^{r_\ell r_{\ell,1} r'_j}, \forall d \in \Gamma/\rho(\ell), H_0(d)^{r_\ell r_{\ell,d} r'_j}),
\end{aligned} \tag{6}$$

$\mathcal{A}_j$ sends $\widetilde{SK_j}$ back to the receiver.

$\mathsf{KeyAggre}(\{\widetilde{SK_\tau}\}_{\tau\in\mathcal{N}}) \rightarrow SK$ is a deterministic key aggregation algorithm. Specifically, to recover the decryption key share SK_τ, the receiver respectively multiplies exponentially with $\{1/r_{i,1}, 1/r_{i,d}\}_{i\in[\ell]}$ on $\widetilde{SK_\tau} = (\widetilde{\varphi_{1,\tau}}, ..., \widetilde{\varphi_{\ell,\tau}})$ and gets $SK_\tau = (\varphi_{1,\tau}, ..., \varphi_{\ell,\tau})$ as follows:

$$\begin{aligned}
SK_\tau = \Big(&(g^{\mu_{1,\tau}} H_0(\rho(1))^{r_1 r'_\tau}, g^{r_1 r'_\tau}, \forall d \in \Gamma/\rho(1), H_0(d)^{r_1 r'_\tau}), ...\\
&(g^{\mu_{\ell,\tau}} H_0(\rho(\ell))^{r_\ell r'_\tau}, g^{r_\ell r'_\tau}, \forall d \in \Gamma/\rho(\ell), H_0(d)^{r_\ell r'_\tau})\Big), \tau \in \mathcal{N}
\end{aligned} \tag{7}$$

The receiver can recover the decryption key SK if he collects any set $\mathcal{N}$ of $t+1$ correct shares $\{SK_\tau\}_{\tau\in\mathcal{N}}$. The decryption key SK is computed as follows:

$$\begin{aligned}
&SK = \{PK, (D_1 = g^{\mu_1}\cdot H_0(\rho(1))^{r_1\sum_{\tau\in\mathcal{N}} r'_\tau\cdot\lambda_\tau}, R_1 = g^{r_1\sum_{\tau\in\mathcal{N}} r'_\tau\cdot\lambda_\tau},\\
&\forall d\in\Gamma/\rho(1), Q_{1,d} = H_0(d)^{r_1\sum_{\tau\in\mathcal{N}} r'_\tau\cdot\lambda_\tau}), ...,\\
&(D_\ell = g^{\mu_\ell}\cdot H_0(\rho(\ell))^{r_\ell\sum_{\tau\in\mathcal{N}} r'_\tau\cdot\lambda_\tau}, R_\ell = g^{r_1\sum_{\tau\in\mathcal{N}} r'_\tau\cdot\lambda_\tau},\\
&\forall d\in\Gamma/\rho(\ell), Q_{\ell,d} = H_0(d)^{r_1\sum_{\tau\in\mathcal{N}} r'_\tau\cdot\lambda_\tau})\}
\end{aligned} \tag{8}$$

$\mathsf{Decrypt}(SK, C)\to M$ is a deterministic decryption algorithm run by the receiver. It takes the decryption key SK and the ciphertext C as inputs, and computes the value $e(g,g)^{\alpha s}$. Let $\Delta = \{x : \exists i\in[\ell], \rho(i) = x\}$, and choose $\{\omega_i\}_{i\in[\ell]}\in\mathbb{Z}_p$ such that $\sum_{i\in[\ell]}\omega_i\cdot W_i = (1,0,...,0)$, it first finishes a pre-processing step on SK and computes the following value:

$$\hat{D}_i = D_i\cdot\prod_{x\in\Delta/\rho(i)} Q_{i,x} = g^{\mu_i}(\prod_{x\in\Delta} H_1(x))^{r_i}. \tag{9}$$

Then, the algorithm does a pre-processing step on C and computes the following values:

$$L = \prod_{x\in\Delta} C_x = \prod_{x\in\Delta} H_1^s(x) = (\prod_{x\in\Delta} H_1(x))^s. \tag{10}$$

After that, the receiver can recover $e(g,g)^{\alpha s}$ by computing:

$$e(\hat{C}_i, \prod_{x\in[\ell]} \hat{D}_i^{\omega_i})/e(\prod_{x\in[\ell]} R_i^{\omega_i}, L) = e(g,g)^{\alpha s}. \tag{11}$$

Finally, the decryption algorithm can divide $e(g,g)^{\alpha s}$ from the ciphertext C and output the message M.

D. Proofs of DKP-ABE-H

1) Proof of Correctness: **Lemma**: *If any each set $\mathcal{N}$ decryption authorities $\{\mathcal{A}_\tau\}_{\tau\in\mathcal{N}}$ provide the correct key shares in the partial secret key generation phase, i.e., $\{\widetilde{SK_\tau}\}_{\tau\in\mathcal{N}}$, $\mathcal{L}$ will surely recover the decryption key SK.*

Proof. Once $\mathcal{L}$ receives $\widetilde{SK} = \{\widetilde{SK_\tau}\}_{\tau\in\mathcal{N}}$ from $t+1$ decryption authorities, he can use $\{1/r_{i,1}, 1/r_{i,d}\}_{i\in[\ell]}$ to compute $\{SK_\tau\}_{\tau\in\mathcal{N}}$. As illustrated in Equation (7), $\mu_{i,\tau} = W_i\cdot\overrightarrow{v_\tau} = W_i\cdot\underbrace{(\alpha_\tau, y_2, ..., y_n)}, i\in[\ell]$. $\mathcal{L}$ multiplies the same column values in $\{\widetilde{SK_\tau}\}_{\tau\in\mathcal{N}}$ together and uses the Lagrange interpolation coefficient λ_τ to multiple exponentially on the corresponding $\widetilde{SK_\tau}$. For instance, the product of the values in the a-th column of $\widetilde{SK_\tau}$ is denoted as follows:

$$\begin{aligned}
&\prod_{\tau\in\mathcal{N}}\left((g^{\mu_{a,\tau}})\cdot H_0(\rho(a))^{r_1 r'_\tau}\right)^{\lambda_\tau}\\
&= g^{\sum_{\tau\in\mathcal{N}}\mu_{a,\tau}\lambda_\tau}\cdot H_0(\rho(a))^{r_1\sum_{\tau\in\mathcal{N}} r'_\tau\cdot\lambda_\tau}\\
&= g^{\sum_{\tau\in\mathcal{N}}\overrightarrow{v_\tau}\cdot W_a\lambda_\tau}\cdot H_0(\rho(a))^{r_1\sum_{\tau\in\mathcal{N}} r'_\tau\cdot\lambda_\tau}\\
&= g^{\sum_{\tau\in\mathcal{N}}(\alpha_\tau, y_2, ..., y_n)\cdot W_a\lambda_\tau}\cdot H_0(\rho(a))^{r_1\sum_{\tau\in\mathcal{N}} r'_\tau\cdot\lambda_\tau}\\
&= g^{\sum_{\tau\in\mathcal{N}}(\lambda_\tau\cdot\alpha_\tau, \lambda_\tau y_2, ..., \lambda_\tau y_n)\cdot W_a}\cdot H_0(\rho(a))^{r_1\sum_{\tau\in\mathcal{N}} r'_\tau\cdot\lambda_\tau}\\
&= g^{(\sum_{\tau\in\mathcal{N}}\lambda_\tau\cdot\alpha_\tau, \sum_{\tau\in\mathcal{N}}\lambda_\tau y_2, ..., \sum_{\tau\in\mathcal{N}}\lambda_\tau y_n)\cdot W_a}\cdot H_0(\rho(a))^{r_1\sum_{\tau\in\mathcal{N}} r'_\tau\cdot\lambda_\tau}\\
&= g^{(\alpha, y'_2, ..., y'_n)\cdot W_a}\cdot H_0(\rho(a))^{r_1\sum_{\tau\in\mathcal{N}} r'_\tau\cdot\lambda_\tau}\\
&= g^{\mu_a}\cdot H_0(\rho(a))^{r_1\sum_{\tau\in\mathcal{N}} r'_\tau\cdot\lambda_\tau},
\end{aligned} \tag{12}$$

where $y'_l = \sum_{\tau\in\mathcal{N}}\lambda_\tau y_l, l\in[2,n]$.

Note that the other values of the exponent are kept the same, i.e., $r_1\sum_{\tau\in\mathcal{N}} r'_\tau\cdot\lambda_\tau$. According to the KP-ABE scheme as in [27], $\mathcal{L}$ can recover the final decryption key.

□

2) Proof of Security:

Theorem 1. *The DKP-ABE-H scheme is selectively secure against chosen-plaintext attacks under the Decisional q-BDHE assumption in the random oracle model.*

Proof. Let $\mathcal{AT}$ be a PPT adversary attacking the proposed CPA security DKP-ABE-H in random oracle model with non-negligible advantage, then we can construct a PPT simulator $\mathcal{SI}$ to break the Decisional q-BDHE assumption. Let q be the maximum number of unique queries made to the random oracle. Before the experiment, the $\mathcal{SI}$ first initializes an empty table T_{RO}.

Init. The adversary $\mathcal{AT}$ outputs an attribute set S^* to challenge ciphertext.

Setup. On receive Decisional q-BDHE tuple

$$(\mathbb{G}, p, g, g^s, g^a, \cdots, g^{(a^q)}, g^{(a^{q+2})}, \cdots, g^{(a^{2q})}, P),$$

for security parameter η, the simulator $\mathcal{SI}$ chooses random numbers $\alpha', z_1, \cdots, z_q\in\mathbb{Z}_p$, and sets $e(g,g)^\alpha = e(g,g)^{\alpha'}\cdot e(g^a, g^{a^q})$. That is defining the MSK $\alpha = \alpha' + a^{q+1}$. Additionally, $\mathcal{SI}$ initializes an empty random oracle table T_{RO}. On any random oracle hash query for attribute x, $\mathcal{SI}$ first looks up whether it is in the table T_{RO}. If so, $\mathcal{SI}$ returns the associated hash value. If x is not in the table, $\mathcal{SI}$ randomly chooses a value $z_x\in\mathbb{Z}_p$, and sets

$$h_x = \begin{cases} g^{z_x} & if\ x\in S^* \\ g^{z_x} g^{a^i} & if\ x\notin S^*, \end{cases} \tag{13}$$

where i is the counter of unique attributes queried to the random oracle. At the end of the setup phase, $\mathcal{SI}$ sets the public parameter as $(\mathbb{G}, p, g, e(g,g)^\alpha)$, and sends it to the adversary $\mathcal{AT}$.

Phase 1. The simulator $\mathcal{SI}$ initializes an empty table T, two empty sets D, E and two integer counters $j = 0$, $l = 0$. Then it responds to $\mathcal{AT}$'s queries as follows:

- Create($\mathbb{A}$): The main challenge of responding the create query is that $\mathcal{SI}$ does not have the master secret key α. On receive an access structure $\mathbb{A}$, $\mathcal{SI}$ sets $j = j + 1$ and parses $\mathbb{A}$ as (W, ρ). Let K be the set of rows where the attributes are in S^* and K' be the set of rows where the attributes are not in S^*. $\mathcal{SI}$ randomly chooses a n-dimensional vector $\overrightarrow{v}$ satisfied $v_1 = 1$ and for all $i \in K, \overrightarrow{v} \cdot W_i = 0$. For all $i \in [1, l]$, $\mathcal{SI}$ first obtains $h_{\rho(i)}$ from the hash random oracle.
  For $i \in K$, set $D_i = R_i = Q_{i,x} = g^0$, where $x \in \Gamma$.
  For $i \in K'$, let $c_i = \overrightarrow{v} \cdot W_i$. $\mathcal{SI}$ sets $R_i = g^{-c_i \cdot a^{(q+1)-\rho(i)}}$ and computes:
$$\begin{aligned} D_i &= g^{c_i \alpha'} \cdot (g^{a^{(q+1)-\rho(i)}})^{-c_i z_i} \\ &= g^{c_i \alpha'} \cdot g^{c_i a^{q+1}} \cdot g^{-c_i a^{q+1}} \cdot (g^{a^{(q+1)-\rho(i)}})^{-c_i z_i} \\ &= g^{\alpha c_i} \cdot g^{-c_i a^{q+1}} \cdot (g^{a^{(q+1)-\rho(i)}})^{-c_i z_i} \\ &= g^{\alpha c_i} \cdot (g^{a^{\rho(i)}})^{r_i} \cdot (g^{z_i})^{r_i} \\ &= g^{\alpha c_i} \cdot h_{\rho(i)}^{r_i}, \end{aligned}$$
  where $r_i = -c_i \cdot a^{(q+1)-\rho(i)}$.
  Then for all $x \in \Gamma/\rho(i)$ $(x \neq \rho(i))$, $\mathcal{SI}$ computes:
$$Q_{i,x} = \begin{cases} (g^{-c_i \cdot a^{(q+1)-\rho(i)}})^{z_x} = h_x^{r_i} & if\ x \in S^*, \\ (g^{-c_i \cdot a^{(q+1)-\rho(i)}})^{z_x} \cdot (g^{-c_i \cdot a^{(q+1)-\rho(i)+x}}) & \\ \quad = (g^{z_x})^{r_i} \cdot (g^{a^x})^{r_i} = h_x^{r_i} & if\ x \notin S^*. \end{cases}$$
  Now, $\mathcal{SI}$ has already constructed a valid secret key. Next, for all $i \in [1, l]$, it re-randomizes the key as follows.
  - Generate random values $r_i', y_2, y_3, \cdots, y_n \in \mathbb{Z}_p$ and set $\lambda' = (0, y_2, y_3, \cdots, y_n) \cdot W_i$.
  - Re-randomize D_i as $D_i' = D_i \cdot g^{\lambda_i' r'}$.
  - Re-randomize $R_i' = R_i \cdot g^{r_i'}$ and $Q_{i,x}' = Q_{i,x} \cdot h_x^{r'}$ for all $x \in \Gamma/\rho(i)$.

  Thus, $(D_i', R_i', \forall d \in \Gamma/\rho(i), Q_{i,d})_{i \in [1,l]}$ is a valid and well-distributed secret key for access structure $\mathbb{A}$.
- CorruptSK(i): If there exists an i-th entry in table T, $\mathcal{SI}$ obtains the entry $(i, \mathbb{A}, SK)$, and sets $D = D \cup \{\mathbb{A}\}$. Then it sends the private key SK to the adversary. Otherwise, it returns $\perp$.
- CorruptPartialMSK(k): If $l \geq t$, it returns $\perp$. Otherwise, $\mathcal{SI}$ sets $l = l + 1, E = E \cup \{\alpha_k\}$ and returns a random value α_k' to the adversary.

Challenge. $\mathcal{AT}$ outputs two messages M_0^* and M_1^* as challenge messages and $\mathcal{SI}$ randomly chooses a bit $b \in \{0, 1\}$. Then $\mathcal{SI}$ constructs and sends the challenge ciphertext
$$C^* = (M_b \cdot P, g^s, \forall x \in S^*, (g^s)^{z_x}).$$

Phase 2. $\mathcal{SI}$ responds the $\mathcal{AT}$'s queries as the same as in Phase 1 except the CorruptSK query for any access structure which S^* satisfied and CorruptPartialMSK query for more than t times are refused.

Guess. $\mathcal{AT}$ outputs the guess b' of b. If $b' = b$, $\mathcal{SI}$ guesses $P = e(g, g)^{a^{(q+1)}s}$, else $\mathcal{SI}$ guesses that P is random.

Thus, $\mathcal{SI}$ answers all $\mathcal{AT}$'s queries with the same distribution as in the $Exp_{\mathcal{AT},\Pi}(\eta, S)$ experiment. Thus, $\mathcal{SI}$ can answer the Decisional q-BDHE problem with the outputs of $\mathcal{AT}$. □

3) Proof of Access Structure Hidden:

Theorem 2. *The DKP-ABE-H scheme perfectly hides the access structure to the authorities.*

Proof. In the SubPolicyHiddenGen algorithm, for each component of φ_i, a random value r_i or $r_{i,j}$ is used to hide the real value the component. Thus, φ_i is indistinguishable with random values to the authorities and the DKP-ABE-H scheme perfectly hides the access structure to the authorities. □

V. Concrete BB-VDF Scheme

The proposed DKP-ABE-H scheme can be utilized to enable secure digital forensics. In this section, we will present the construction of BB-VDF based on DKP-ABE-H and blockchain. Specifically, section V-A outlines the overview and depicts the blockchain based forensics state machine. Section V-B introduces the concrete scheme and section V-C analyzes the security of BB-VDF.

A. Overview of BB-VDF Scheme

In the concrete scheme, we use the vehicles (i.e., $\mathcal{S}$) as illustration to denote how the forensics data are generated and collected[4]. A forensics daemon in vehicles collects the real-time data from different sensors (e.g., the latest vehicle operations), and submits the related data to $\mathcal{D}$ and $\mathcal{B}$ periodically. In doing so, data integrity can be verified with the tamper-resistant transaction records in the blockchain.

In terms of the forensics phase, $\mathcal{L}$ first generates a warrant which contains the metadata and detailed information (i.e., ① in Fig.1). The metadata is published in $\mathcal{B}$ while the detail information which specifies the forensics data range (i.e, attributes) is kept secret from any unauthorized parties. When $\mathcal{L}$ is allowed to conduct the forensics investigation by $\mathcal{C}$, it will step into phase 2. Specifically, we mainly focus on the processes of data collection and data reporting (i.e., ②, ⑤ in Fig.1). During the collection phase, $\mathcal{L}$ can collect forensics data from 1) the vehicles $\mathcal{S}$ and 2) the data storage $\mathcal{D}$. To obtain real-time data from the vehicle, $\mathcal{L}$ can use the authorized secret key to generate a signature, the forensics daemon will verify the validation of the signature and send a diagnostic command to each ECU, and $\mathcal{L}$ can get the real-time data. To obtain historical forensics data, $\mathcal{L}$ needs to require the decryption key from the authorities, and then downloads the data from $\mathcal{D}$. During the data analysis, our scheme follows the main idea of block4forensics [11] that integrates different kinds of data to conduct the analysis. Here, we introduce two key techniques 1) state machine and 2) blockchain oracle that are used to enhance the efficiency and security of BB-VDF.

State Machine: The BB-VDF scheme is centered around a smart contract which depicts the vehicle forensics process as a state machine. According to the forensics life cycle, we model each procedure as a state in FSM and represent them in smart contracts. To realize a more secure VDF, state transition follows the *verification-then-forwarding* model which requires

[4]Our proposed scheme can be extended to support other data sources with a little modification.

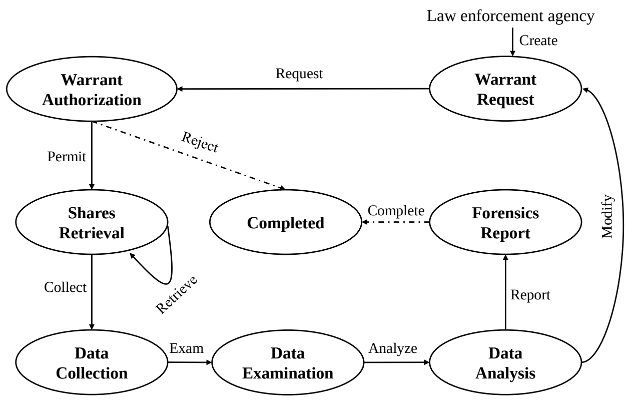


Fig. 3. The state machine model for vehicle digital forensics.

state change to be accomplished with a transaction in the blockchain and be publicly verified by the blockchain nodes.

As illustrated in Fig. 3, there are eight states in FSM: Warrant Request, Warrant Authorization, Shares Retrieval, Data Collection, Data Examination, Data Analysis, Forensics Report, and Completed. We define the blockchain transaction as an "Event" that triggers the state transition in FSM. The inputs for different state transition events are derived from different parties. More precisely, the process that a state σ_1 transits into another state σ_2 based on a transaction TX can be denoted as: $\sigma_1 \xrightarrow{TX} \sigma_2$. The initial state Warrant Request is set by $\mathcal{L}$, which is to initialize a warrant. The following state transition can happen only if $\mathcal{C}$ authorizes the request, otherwise the state will be transited to Completed.

Blockchain Oracle: As mentioned in the introduction, $\mathcal{C}$ may forget to track the states of the warrants which are actually expired, while other entities may not realize the expiration and still maintain cooperation, which is apparently unsatisfied with the general forensics process. To avoid this and ensure each involved entity has a clear understanding on the state of the forensics investigation, we leverage the off-the-shelf blockchain oracle [35] to schedule specific tasks and pass the external information to smart contracts for execution. The blockchain oracle is initialized and authenticated by $\mathcal{C}$. That is, the oracle relays the legal instructions of $\mathcal{C}$ to the smart contract. It is mainly responsible for scheduling transactions to be executed after a certain time. By doing so, some of the state transitions are accomplished by specifying the pre-prescribed events which follow the standard forensics investigation. Furthermore, if an "Event" is not triggered within a predefined time, then the corresponding entities will be informed by the blockchain oracle, or the state could be automatically transited to Completed under the setting.

B. Concrete Scheme

Here, we present the concrete BB-VDF scheme to demonstrate the utility of DKP-ABE-H and blockchain in the forensics scenario.

1) System Initialization: During the initialization phase, the court $\mathcal{C}$ and the authorities $\mathcal{A}$ collaboratively initialize the whole system with the algorithm Setup$(1^\eta, t, n)$. More concretely, let H_1, H_2 be two cryptographic hash functions: $H_0, H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p$, $H_2 : \mathbb{G}_T \rightarrow \{0,1\}^\kappa$, where κ is the length of the symmetric encryption key, e.g., 128 bits. The algorithm specifies the master public parameters $PK = (\mathbb{G}, p, g, e(g,g)^\alpha, H_0, H_1, H_2)$ and the master secret key $MSK = \{\alpha \in \mathbb{Z}_p\}$. Besides, the involved parties generate their key pairs (i.e., public key and private key). All public keys are required to be authenticated by $\mathcal{C}$ and published in the blockchain that the public can verify the validity of a message[5].

2) Digital Forensics Data Generation: During the data generation phase, data sources periodically submit the data to $\mathcal{D}$. The period can be set flexibly, e.g., one hour or an half hour for the consideration of real-time forensics. More precisely, a data submitted by $\mathcal{S}_i$ is denoted as $d_{id,type,t}$, where these subscript variables are essentially defined as *attributes*, id refers to the identity of $\mathcal{S}_i$ (i.e., $K^p_{\mathcal{S}_i}$), $type$ refers to data type (e.g., steering wheel, brake, seat belt), and t refers to timestamp. For simplicity, we assume that vehicles submit the data with an integer timestamp, such as tagging a data $d_{id,type,t}$ with the metadata ("*id*: Alice", "*type*: steering wheel", "*t*: 2020/07/22 22:00").

The forensics daemon in $\mathcal{S}_i$ interacts with the ECUs and collects data $\beta_{t_1} = \{d_{id,type_1,t_1}, d_{id,type_2,t_1}, ...\}$ at time t_1. These data are encrypted with a hybrid encryption method and sent to $\mathcal{D}$. Meanwhile, $\mathcal{S}_i$ submits a transaction for each data $d_{id,type,t}$ which contains the metadata (e.g., hash value, timestamp). To reduce the on-chain transactions, $\mathcal{S}_i$ can compress a set of consecutive data (e.g., $\beta_{t_1 \sim t_\varsigma} = \{\beta_{t_1}, ..., \beta_{t_\varsigma}\}$) into a Merkle Hash tree and store the Merkle root to a transaction. Specifically, the process of the data generation using Encrypt(M, PK, S) can be depicted as follows:

- At the beginning, to encrypt $d_{id,type,t_1}$, $\mathcal{S}_i$ randomly generates two numbers $s_1 \in \mathbb{Z}_p, \varepsilon_1 \in \mathbb{G}_T$ and computes the hash value $k_1 = H_2(\varepsilon_1)$. Then, $\mathcal{S}_i$ uses k_1 to encrypt the data $d_{id,type,t_1}$ with symmetric encryption algorithm (e.g., AES-128) and computes the values as follows:

$$\begin{aligned} &k_1 = H_2(\varepsilon_1), \hat{C} = g^{s_1}, M_{id,type,t} = Enc_{k_1}(d_{id,type,t_1}), \\ &H_{M_{id,type,t_1}} = H_1(M_{id,type,t_1}), C = \varepsilon_1 \cdot e(g,g)^{\alpha s_1}, \\ &\{C_x = H_1(x)^{s_1}\}_{x\in\{0,1\}^*}, \theta_1 = H_1(id||type||t_1), \end{aligned} \tag{14}$$

  where θ_1 is defined as the corresponding *download link* of $d_{id,type,t_1}$. The encrypted data $\{\theta_1 : (M_{id,type,t_1}, C, \hat{C}, \{C_x\}_{x\in\{0,1\}^*})\}$ is stored in $\mathcal{D}$.
- After collecting some a set of data (e.g., from t_1 to t_ς), the forensics daemon computes the Merkle root $root$ using the consecutive $m \times \varsigma$ ciphertexts as:

$$root = MerkleRoot(H_{M_{id,type_1,t_1}}, ..., H_{M_{id,type_m,t_\varsigma}}) \tag{15}$$

.

- Vehicle $\mathcal{S}_i$ signs the data using the private key and generates a transaction $TX^1_{\mathcal{S}_i}$:

$$TX^1_{\mathcal{S}_i} = [root, (\theta_1, ..., \theta_{m\times\varsigma}), Ti]_{K^s_{\mathcal{S}_i}}, \tag{16}$$

[5]The identity management and authentication is not depicted here, which is not the key point in this paper.

where Ti refers to the generation timestamp of the transaction.

3) Warrant Request: Warrant request is conducted by following the $\mathsf{SubPolicyHiddenGen}(\mathbb{A})$ algorithm in DKP-ABE-H. To guarantee the auditability and validation of the forensics process, the algorithm execution is achieved with an interactive process between the agency $\mathcal{L}$ and the court $\mathcal{C}$.

More concretely, in case of accidents or suspicious behavior, $\mathcal{L}$ requests a warrant from $\mathcal{C}$ by submitting a transaction to create a new state machine. The warrant contains a short description des (i.e., metadata) and a policy $\mathbb{A}$ (i.e., an access structure including target identity, data type, and time). $\mathcal{L}$ sends the warrant to $\mathcal{C}$ through the secure channel. If $\mathcal{C}$ approves the investigation, he signs on the warrant and generates some parameters which are used to recover the decryption key. $\mathcal{L}$ is allowed to request forensics data from multiple data sources, different data types and timestamp by constructing a comprehensive access $\mathbb{A}$, e.g., *("*id*: David" OR "*id*: Carl") AND (2020/06/21 22:00 $\leq$"*time*"$\leq$ 2020/06/23 22:00)*. The interactive processes of warrant request can be depicted as follows:

- $\mathcal{L}$ sends a warrant $req = (\mathbb{A}, des)$ to $\mathcal{C}$ and submits a transaction to the blockchain.

$$TX_{\mathcal{L}}^2 = [des, H_{req}, Ti]_{K_{\mathcal{L}}^s}, \tag{17}$$

  where $H_{req} = H_0(req)$ is an identifier of the warrant in $\mathcal{B}$. The public can audit the legitimacy of an investigation through the identifier. At this point, an FSM instance is created in the smart contract as: $* \xrightarrow{TX_{\mathcal{L}}^2} \mathsf{Warrant\ Request}$.
- $\mathcal{C}$ receives the structure $\mathbb{A}$ and evaluates whether to approve the investigation request. If no, $\mathcal{C}$ submits a transaction to terminate the FSM instance (i.e., prompts the FSM to $\mathsf{Completed}$). Otherwise, $\mathcal{C}$ selects ℓ sets of random numbers $r_1, ..., r_\ell \leftarrow \mathbb{Z}_p$ and generates the following values:

$$\begin{aligned} &\xi_1 = (H_1(\rho(1))^{r_1}, g^{r_1}, \forall d \in \Gamma/\rho(1), H_1(d)^{r_1}),\\ &...\\ &\xi_\ell = (H_0(\rho(\ell))^{r_\ell}, g^{r_\ell}, \forall d \in \Gamma/\rho(\ell), H_0(d)^{r_\ell}),\\ &res = (\xi_1, \xi_2, ..., \xi_\ell), \end{aligned} \tag{18}$$

  where res is a response to be sent back to $\mathcal{L}$ through the secure channel.
- To prevent $\mathcal{L}$ from abusing his power, $\mathcal{C}$ submits a transaction with a digital signature on the data $(H_1(\rho(1))^{r_1}, ..., H_1(\rho(\ell))^{r_\ell})$, allowing the authorities and the public to confirm that a decryption key request from $\mathcal{L}$ has been authorized by $\mathcal{C}$.

$$TX_{\mathcal{C}}^3 = [H_{req}, H_1(res), H_0(\rho(1))^{r_1}, ..., H_0(\rho(\ell))^{r_\ell}, Ti]_{K_{\mathcal{C}}^s}. \tag{19}$$

Once $\mathcal{B}$ confirms the transaction $TX_{\mathcal{C}}^3$, the state of the FSM instance will be transited: $\mathsf{Warrant\ Request} \xrightarrow{TX_{\mathcal{C}}^3} \mathsf{Warrant\ Authorization}$.

4) Auditable Data Collection: In the data collection phase, $\mathcal{L}$ retrieves a decryption key from the majority of the authorities. Specifically, $\mathcal{L}$ computes a set of blinded values on res and sends them to multiple authorities who can provide a share of the decryption key according to their judgement. $\mathcal{L}$ can recover the decryption key when receiving $\mathcal{N}$ of $t+1$ decryption key shares from $\mathcal{A}$.

In particular, to preserve the privacy of investigation, the attributions $\rho(1), ..., \rho(\ell)$ are not revealed. To achieve this, $\mathcal{L}$ can choose a set of random numbers $r_{\epsilon_1, d} \in \mathbb{Z}_p, \epsilon_1 \in [1, \ell]$ and computes the blinded values on res. As illustrated in equation (4), the parameters $\{\varphi_1, ..., \varphi_\ell\}$ are sent to each authority. After that, $\mathcal{L}$ sends a transaction to the blockchain to apply for the decryption key. At this point, the state of the FSM instance is transited: $\mathsf{Warrant\ Authorization} \xrightarrow{TX_{\mathcal{L}}^4} \mathsf{Shares\ Retrieval}$.

$$\begin{aligned} TX_{\mathcal{L}}^4 = [&H_{req}, H_0(\rho(1))^{r_1}, ..., H_0(\rho(\ell))^{r_\ell},\\ &H_1(\varphi_1), ..., H_1(\varphi_\ell), Ti]_{K_{\mathcal{L}}^s}. \end{aligned} \tag{20}$$

Each authority checks whether a request is valid by confirming that the requested decryption key is corresponding to the authorized warrant (by $TX_{\mathcal{C}}^3$ and $TX_{\mathcal{L}}^5$). If yes, it helps to recover the decryption key with the $\mathsf{SubKeyGen}(\{\varphi_j\}_{j\in[\ell]}, \alpha_i)$ algorithm. Specially, to prevent others from learning $g^{\mu_{a,j}}$, $\mathcal{A}_j$ can choose a random number $r_j' \in \mathbb{Z}_p$ and compute the secret share $\widetilde{SK_j} = (\widetilde{\varphi_{1,j}}, ..., \widetilde{\varphi_{\ell,j}})$. In the meanwhile, to allow the public to verify that the provided shares are indeed computed according to the authorized warrant, an authority leverages the zero knowledge proof technique [36] to approve that $(\widetilde{\varphi_{1,j}}, ..., \widetilde{\varphi_{\ell,j}})$ are indeed computed based on the authorized parameters $res = (\xi_1, ..., \xi_\ell)$. Each authority is required to submit a transaction to prove the validity of share-providing. For instance, a transaction submitted by $\mathcal{A}_j$ can be shown as follows:

$$TX_{\mathcal{A}_j}^5 = [H_{req}, g^{\mu_{1,j}} H_0(\rho(1))^{r_1 r_j'}, ..., g^{\mu_{\ell,j}} H_0(\rho(\ell))^{r_\ell r_j'}, Ti]_{K_{\mathcal{A}_j}^s}. \tag{21}$$

$\mathcal{L}$ can recover the decryption key SK if he collects any set $\mathcal{N}$ of $t+1$ correct shares $\{\widetilde{SK_\tau}\}_{\tau\in\mathcal{N}}$. Suppose that more than t authorities have provided the shares, the state of the FSM instance is automatically transited: $\mathsf{Shares\ Retrieval} \xrightarrow{\{TX_{\mathcal{A}_\tau}^5\}_{\tau\in\mathcal{N}}} \mathsf{Data\ Collection}$.

Then, $\mathcal{L}$ can retrieve the related data from $\mathcal{D}$ after recovering the decryption key SK. Specially, to prevent $\mathcal{D}$ from learning the value of the attributes, we can leverage the private information retrieval (PIR) scheme to retrieve the data from $\mathcal{D}$ [37]. By doing so, the process of data retrieval does not expose the information whose data have been downloaded. $\mathcal{L}$ can compute the data decryption key and obtain the corresponding forensics data. Meanwhile, $\mathcal{L}$ submits a transaction to prompt FSM into new state as: $\mathsf{Data\ Collection} \xrightarrow{TX_{\mathcal{L}}^6} \mathsf{Data\ Examination}$ by $\mathcal{L}$.

$$TX_{\mathcal{L}}^6 = [H_{req}, H_1(H_{D_{id,type,t}}), Ti]_{K_{\mathcal{L}}^s} \tag{22}$$

5) Data Examination and Analysis: In this phase, $\mathcal{L}$ comprehensively conducts systematic search of the collected data, identifying the potential evidence. Firstly, $\mathcal{L}$ needs to confirm that the integrity of an evidence is preserved, which can be achieved by using the Merkle tree proof verification algorithm [38]. Then, the software and hardware forensics methods can be used to extract valuable evidences [7]. After that, the state machine is transited as Data Examination$\xrightarrow{TX_{\mathcal{L}}^{7}}$Data Analysis by $\mathcal{L}$. Data analysis is to further analyze the evidence collected in the data examination phase. It is worth noting that if $\mathcal{L}$ requires more data, he needs to reapply for a new warrant by following the procedures. Finally, the state machine will be transited into Forensics Report (denoted as $TX_{\mathcal{L}}^{8}$) by $\mathcal{L}$.

6) Automated Vehicle Forensics Reporting: With the time-scheduling service based on the blockchain oracle, BB-VDF allows a final forensics report to be sent to different parties automatically. For instance, when a traffic accident happens, the final report is required to be sent to the drivers and insurance company simultaneously, which can help to make a judgement on indemnity. For the sake of future investigation, the digest of reports is recorded in $\mathcal{B}$ (denoted as $TX_{\mathcal{L}}^{9}$). If both $\mathcal{C}$ and $\mathcal{L}$ have confirmed the report, the state machine will be transited into Complete (denotes as $TX_{\mathcal{C}}^{10}$, $TX_{\mathcal{L}}^{10}$, respectively).

As shown in Fig. 4, we present the design of the BB-VDF contract for the state machine transition, where each function is called by the corresponding party. In the "Complete" phase, when $\mathcal{L}$ accomplishes an investigation, a multi-signature transaction is required from $\mathcal{L}$ and $\mathcal{C}$ to terminate the FSM instance.

```
BB-VDF Contract
On receive ("Request", $des$, $H_{req}$) from $\mathcal{L}$.
    create st;
    set st := WARRANT REQUEST;
    put warrrantPool[$H_{req}$] = (des, msg.sender, block.time);
    initialize sharesNum = 0; ▷ the collected key shares number;
    broadcast ("An investigation request has been created.");
    ▷ e.g., by Event mechanism on Ethereum
On receive ("Permit", $H_{req}$, $H_0(\rho(1))^{r_1}$, ..., $H_0(\rho(\ell))^{r_\ell}$) from $\mathcal{C}$.
    ▷ audit the warrant request off-chain.
    assert st = WARRANT REQUEST;
    parse $req$ and check the validity as in Equation (12);
    put warrrantPool[$H_{req}$].keyPara = ($H_0(\rho(1))^{r_1}$, ..., $H_0(\rho(\ell))^{r_\ell}$);
    set st := WARRANT AUTHORIZATION;
    broadcast ("An investigation has been permitted by the court.");
On receive ("Reject", $H_{req}$) from $\mathcal{C}$.
    ▷ clean intermediate variables.
    delete warrrantPool[$H_{req}$];
    set st := COMPLETE;
    broadcast ("An investigation has been rejected by the court.");
On receive ("Store", $H_{req}$, $H_1(\varphi_1)$, ..., $H_1(\varphi_\ell)$) from $\mathcal{L}$.
    assert st = WARRANT AUTHORIZATION;
    put warrrantPool[$H_{req}$].keyBlindPara = ($H_1(\varphi_1)$, ..., $H_1(\varphi_\ell)$);
    broadcast ("The law enforcement submits the blinded values.");
On receive ("Retrieve", $H_{req}, \pi_1, g^{\mu_{1,j}} H_0(\rho(1))^{r_1 r'}$, ...,
    $g^{\mu_{\ell,j}} H_0(\rho(\ell))^{r_\ell r'}$) from $A_j$.
    assert st = WARRANT AUTHORIZATION;
    put warrrantPool[$H_{req}$].sharesPara = ($\pi_1, g^{\mu_{1,j}} H_0(\rho(1))^{r_1 r'}$, ...,
    $g^{\mu_{\ell,j}} H_0(\rho(\ell))^{r_\ell r'}$);
    sharesNum += 1;
    ▷ check whether $t+1$ shares have been collected;
    if sharesNum $\geq t+1$
      set st := DATA COLLECTION;
      broadcast ("The investigator collects enough shares.");
    broadcast ("The $A_j$ submits the decryption key shares.");
On receive ("Collect", $H_{req}$, $H_1(H_{D_{id,type,t}})$) from $\mathcal{L}$.
    assert st = DATA COLLECTION;
    put warrrantPool[$H_{req}$].dataDigest=$H_1(H_{D_{id,type,t}})$;
    assert st = DATA EXAMINATION;
    broadcast ("The investigator has collected the forensics data.");
▷ NOTE: The process of data examination and analysis are skipped.
On receive ("Report", $H_{req}$, $H_1(report)$) from $\mathcal{L}$.
    assert st = DATA ANALYSIS;
    put warrrantPool[$H_{req}$].report=$H_1(report)$;
    set st := FORENSICS REPORT;
    broadcast ("The investigator starts to examine the forensics data.");
On receive ("Complete", $H_{req}$, $multiSignature$) from $\mathcal{L}$ and $\mathcal{C}$.
    verify multiSignature;
    assert st = FORENSICS REPORT;
    set st := COMPLETE;
    broadcast ("The investigation has been accomplished.");
```

Fig. 4. BB-VDF contract for the state machine.

C. Security Analysis of BB-VDF

In this section, we will elaborate the security analysis of the proposed scheme.

1) Accountability: The accountability of the scheme includes three aspects: process audit, verifiability, and authorization.

a) Complete Process Audit: As illustrated in Fig. 3, we model the process of VDF as a complete FSM, in which each state transition is completed with a blockchain transaction. The data evidence which are helpful for the forensics are stored in a transaction that no one can repudiate. Specifically, the verification algorithms have been predefined in smart contracts. As long as the underlying blockchain is secure, it is impossible for malicious attackers to prompt the state machine transition without being authorized or verified. Note that there are 8 states of the state machine and at least $(10+t)$ transactions need be written into the blockchain (if $\mathcal{C}$ authorizes the warrant request), i.e., $TX_{\mathcal{L}}^{2}, TX_{\mathcal{C}}^{3}, TX_{\mathcal{L}}^{4}, \{TX_{\mathcal{A}_\tau}^{5}\}_{\tau \in \mathcal{N}}$ ($t+1$ transactions), $TX_{\mathcal{L}}^{6}, ..., Tx_{\mathcal{C}}^{10}, TX_{\mathcal{L}}^{10}$. These data (similar to *audit logs* in [15]) on a specific warrant are recorded in blockchain with tamper-resistance and traceability. Thus, the BB-VDF scheme is able to achieve complete process audit.

b) Public Verifiability: The whole operations of a forensics process are recorded in the permissioned-based blockchain. We assume that the on-chain data are not confidential and are public. Thus, the process of state transitions for any warrant are transparent, allowing the unauthorized or illegal behaviors under the public supervision. In addition, as long as the security assumption holds, $\mathcal{L}$ can not acquire more data than being actually authorized by $\mathcal{C}$. Also, the public could verify whether a share is computed based on the authorized warrant and its validation can be checked securely with zero knowledge proof.

c) Authorization: It is straightforward that authorization can be achieved by the *verification-then-forwarding* state machine. Before the data collection, it is necessary to submit $\mathcal{C}$'s digital signature for authorizing the warrant request to the BB-VDF smart contracts, which can ensure the legitimacy of the investigation. Furthermore, during the verification of a warrant, $\mathcal{C}$ will verify the validation of requested access policy $\mathbb{A} = (W, \rho)$, and generate the intermediate parameters (i.e., $\big(H_0(\rho(1))^{r_1}, ..., H_0(\rho(\ell))^{r_\ell}\big)$) which will be used to recover SK. During the secret shares collection, the authority will check whether the validation of the values which are authorized by $\mathcal{C}$. If not, the authority can refuse to provide the secret share.

2) Privacy Preservation: The privacy preservation of the scheme includes two aspects: the confidentiality of warrant and forensics data.

a) Confidentiality of Warrant: The privacy of a warrant can be kept secret against malicious users in our scheme. Specifically, during the warrant authorization, $\mathcal{L}$ sends the warrant details req to $\mathcal{C}$ through the secure channel without exposing it to others. Besides, the attributions $\big(\rho(1), ..., \rho(\ell)\big)$ in $TX^3_{\mathcal{C}}$, $TX^4_{\mathcal{L}}$ and $\{TX^5_{\mathcal{A}_\tau}\}_{\tau\in\mathcal{N}}$ are blinded using the hash function $H_0(\cdot)$ and random numbers $(r_1, ..., r_\ell), \{r'_\tau\}_{\tau\in\mathcal{N}}$. Thus, even though the malicious users could enumerate the values of $H_0(\rho(x)), x = (\rho(1), ..., \rho(\ell))$, as long as $\mathcal{L}$ preserves these random numbers, others can not learn about the attributions, which is based on the intractability of discrete logarithm problem (DLP).

b) Confidentiality of Forensics Data: In our design, $\mathcal{L}^*$ can not obtain the plaintext data directly from $\mathcal{S}$, he needs to be authorized by $\mathcal{C}$ to obtain the decryption key from the authorities. The legitimacy for requiring the decryption key can be audited in blockchain. There are two ways to obtain the decryption key for the honest-but-curious $\mathcal{C}$: first, 1) $\mathcal{C}$ can collude with $\mathcal{L}$ who provides the decryption key, which is contrary to the security assumption. Second, 2) $\mathcal{C}$ colludes with the authorities $\mathcal{A} = \{\mathcal{A}_1, ..., \mathcal{A}_n\}$. However, we assume that at least more than t authorities are honest. According to the standard DKG protocol [26], it is impossible for $\mathcal{C}$ to restructure the decryption key SK with less than $t+1$ shares. Herein, our scheme can resist against collusion attack between $\mathcal{C}$ and $\mathcal{A}$. Furthermore, since the random numbers $r_{\epsilon_1,d}$ are kept secret, it is impossible for malicious users to recover the final secret shares $\{SK_\tau\}_{\tau\in\mathcal{N}}$. As for $\mathcal{A}$, it is also obvious for him to recover the decryption key under the standard *t-of-n* threshold assumptions of secure DKG.

3) Data Security: The data security of the scheme includes three aspects: availability, integrity, and unforgeability.

a) Availability: Availability in our scheme means that the parties are able to conduct or participate in the vehicular digital forensics at anywhere and anytime. Put simply, it mainly includes two aspects: the availability of data and forensics service. First, the data is stored in the distributed data storage, enabling any party who has the decryption key can download the data. Second, the underlying blockchain is a decentralized architecture which can resist denial of service (DoS) attacks, and is exempt from SPoF/C. Thus, involved parties are able to access the blockchain-based system with high security to prompt the completion of an investigation on-chain.

b) Integrity: It is fairly straightforward that integrity can be guaranteed with the open blockchain. The digest information (i.e., the hash value of the forensics data) of data records are recorded in $\mathcal{B}$ using the Merkle hash tree before an investigation, and no one can tamper with these records. In addition, if the data are modified off chain, it can be easily detected with the hash values recorded in $\mathcal{B}$.

c) Unforgeability: Since each data record is recorded with a digital signature, it is apparent for the unforgeability. Specifically, we consider two scenarios in which malicious attackers could forge evidence to disrupt the normal forensics process in the BB-VDF. First, $\mathcal{L}^*$ might use the expired warrant or create a forged warrant to acquire a decryption key from the $\mathcal{A}$. However, the metadata of the warrant should be published in the BB-VDF contracts which will automatically check the timestamp on whether the warrant has expired. As long as no more than $t+1$ authorities provide correct shares, it is impossible for $\mathcal{L}^*$ to get the decryption key. Second, an unauthorized person may personate $\mathcal{L}$ to acquire data. However, due to the strict audit of $\mathcal{C}$, $\mathcal{A}$ and the permissioned blockchain nodes, the probability of this attack is negligible.

VI. Implementation and Evaluation Results

In this section, we present the implementation results of the proposed scheme which focus on three aspects: (i) communication and computation costs analysis, (ii) experiments for the time performance analysis off-chain, and (iii) experiments for the transaction time performance analysis on-chain. Specifically, we implement roughly 16.7k lines of Java, Javascript and Solidity.

A. Simulation Design

We implement the cryptography algorithms on the DKG and KP-ABE protocol off-chain based on JPBC which is the PBC pairing-based cryptography library. H_0 is the JPBC built-in implementation of SHA256. H_1, H_2 are hash functions that convert the output of H_0 to $\mathbb{G}_1$. We set the system parameters based on the type A prime-order bilinear groups with 160_bit $\mathbb{Z}_p$ and 512_bit $\mathbb{G}_1$. Specifically, we implement the cryptographic parts off-chain using *CloudCrypto* which is written in the Java[6]. CloudCrypto has supported the universe KP-ABE protocol [39]. We modify the source code to enable it to support decentralized key generation and aggregation.

The experiments are conducted on a personal computer ("Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz", 4GB RAM). We design the DKG protocol based on 5 authorities (*3-of-5*) to evaluate the performance of the communication and computation costs. Following the standard KP-ABE protocol, we summarize the investigation process as four stages: Setup, KeyGen, Encryption, and Decryption. The Setup phase includes the initialization of the DKG and KP-ABE protocols. Specifically, the KeyGen phase includes the secret shares retrieval and secret key (i.e., SK) recovery. We count the time performance on Encryption and Decryption under the hybrid encryption method that mainly records the symmetric key encryption and decryption. The time cost of encryption and decryption of data using symmetric algorithms are straightforward, which will not be analyzed here.

To show the practicability of the proposed scheme, we implement our state machine contract (the BB-VDF contracts) in the official Ethereum public test network *Rinkeby*[7]. The

[6]https://github.com/liuweiran900217/CloudCrypto
[7]https://rinkeby.etherscan.io/

TABLE II
COMMUNICATION COST IN SETUP, ENCRYPTION, KEY GENERATION,AND DECRYPTION.

Phases	Communication Cost
Setup	$n^2(t \cdot \lfloor \mathbb{G}_1 \rfloor + 2\lfloor \mathbb{Z}_p \rfloor)$
Encryption	$\lfloor Enc(M) \rfloor + m\varsigma\lfloor H_1 \rfloor + \lfloor \mathbb{G}_1 \rfloor + \lfloor \mathbb{G}_T \rfloor$
KeyGen	$(n+t) \cdot (\ell^2\lfloor H_0 \rfloor + \ell\lfloor \mathbb{G}_1 \rfloor) + \lfloor \mathbb{A} \rfloor + 2\ell t\lfloor H_0 \rfloor + \ell\lfloor H_1 \rfloor$
Decryption	$\lfloor Enc(M) \rfloor + \lfloor H_1 \rfloor$

[1] t is the number of authorities who submit the shares.
[2] The communication cost of PIR is not included.

BB-VDF contracts are mainly composed of 14 sub-contracts. We evaluate the performance of our scheme by considering the network latency and transaction synchronous under the public blockchain network. We download *MetaMask* which allows us to connect to the chosen Rinkeby test network and run the smart contracts in the browser without running a full blockchain node[8]. The maximum gas limit is set as the same for different transactions.

B. Computation and Communication Cost Analysis

Here, we discuss the computation and communication costs incurred by the proposed scheme. We mainly summarize the expensive computation and communication costs in the forensics process. Different notations are utilized to denote the operations in the corresponding group. Specifically, $\mathbb{G}_k$ denotes modular exponentiation in the corresponding group, i.e., $\mathbb{G}_1, \mathbb{G}_T$. H_k denotes the specific hash function, i.e., H_0, H_1 and H_2. $\mathbb{G}_k^m$ refers to the multiple modular exponentiation in group $\mathbb{G}_k$ with m bases. P denotes the bilinear pairing in KP-ABE. ℓ is the attribute number in the access policy $\mathbb{A}$. $\mathcal{N}$ denotes the number of authorities who participate in the secret share recovery. As for the vehicles, the main computation cost is mainly the data decryption, i.e., (Enc + $\mathbb{G}_T^2$ + $\ell \cdot \mathbb{G}_1 + G_1 + H_0 + 2H_2$). As for $\mathcal{L}$, the main computation cost lies in the data collection phase. Specially, after $\mathcal{C}$ returns res to $\mathcal{L}$, $\mathcal{L}$ blinds these values and computes the final decryption key, i.e., $(\ell+1) \cdot H_0 + \ell \cdot \big((\ell+1) \cdot \mathbb{G}_1 + \cdot 2\mathbb{G}_1 + (t+3) \cdot P + \mathbb{G}_1^{2\mathcal{N}} + 2 \cdot \mathcal{N} \cdot \mathbb{G}_1\big) + 2\mathbb{G}_1^{\mathcal{N}}$. As for $\mathcal{C}$, he authorizes a warrant and generates the initial parameters, i.e., $\ell \cdot \big((\ell+1) \cdot \mathbb{G}_1 + H_1\big)$. Besides, for a single authority, the main computation cost is to calculate $\widetilde{SK}_j$, i.e., $\ell \cdot \mathbb{G}_1^2 + \ell \cdot (\ell+1) \cdot \mathbb{G}_1$.

In terms of communication cost analysis, we use $\lfloor \mathbb{G}_x \rfloor$ to denote an element length in group $\mathbb{G}_x$. $\lfloor \mathbb{A} \rfloor$ refers to the size of the warrant. $\lfloor \mathbb{Z}_p \rfloor$ is used to refer to an element length in $\mathbb{Z}_p^*$. $\lfloor H_x \rfloor$ denotes the output length of different hash functions (e.g., H_0, H_1, H_2). Let $\lfloor M \rfloor$ be the size of the forensics data. The corresponding communication cost on different phases are shown in TABLE II.

C. On-chain Performance Evaluation

We record the time consumption for each transaction involved in the execution of the warrant. There are 10 types of transactions involved in the proposed scheme. As shown in Fig. 5, the average confirmation time for a specific transaction is about 32.89s. Namely, each transaction takes about 2 blocks of time to be finally confirmed in the Rinkeby (a new block is generated in every 15s). The confirmation time is not related to the size of the transaction but the number of transactions in that specific time. Generally, the time consumption of transaction confirmation in the blockchain is acceptable. In order to achieve fast transaction confirmation, we can dynamically adjust the gas limitation for a block or decrease the consensus time in the customized blockchain network.

[8]https://metamask.io/

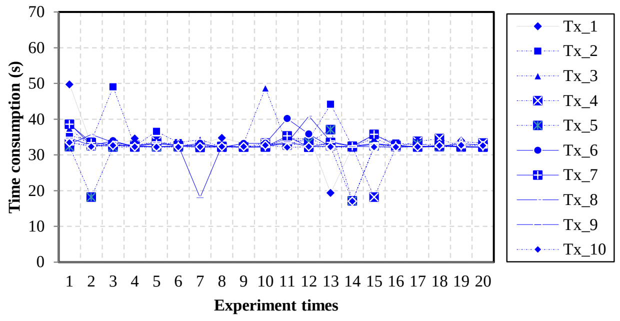
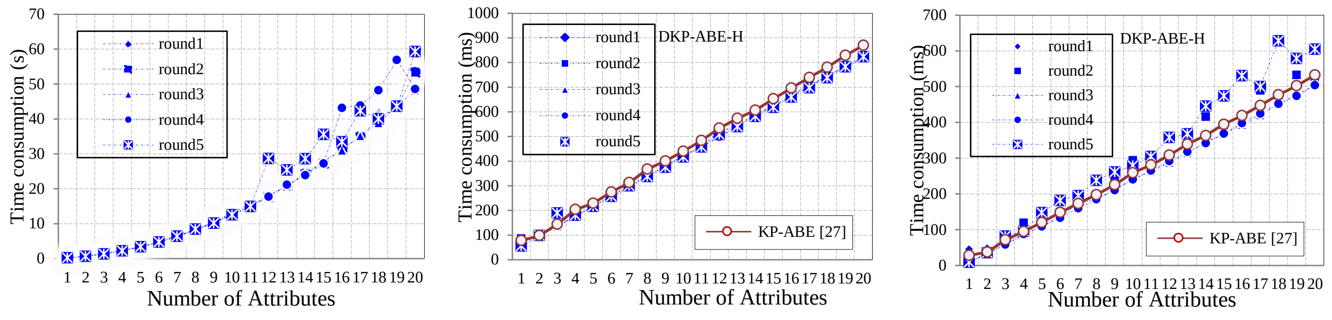


Fig. 5. The experiment results in reaching a consensus on verifying the validation of a transaction.

D. Off-chain Performance Evaluation

We evaluate the off-chain time performance for different phases. Specifically, we conduct the experiments for 5 rounds and compare our customized D-KP-ABE with [27]. To check the impacts on the time performance under attribute number, we conduct experiments with a different number of attributes in each round, i.e., from 1 to 20 attributes. We only use "AND" in the access policy. According to the statistics, the average time consumption for the setup phase is about 35.91ms. Fig. 6 shows the time consumption versus the number of attributes. To be specific, the computation time costs for different phases are approximately linear with the increasing of the attributes. The key generation takes the most of the time compared with the other three phases. The time costs including the partial key generation and the final secret key SK aggregation. More concretely, $\mathcal{C}$ needs to generate the parameters $\xi_1, ..., \xi_\ell$, and $\mathcal{L}$ computes the blinded values of these parameters. In addition, $\mathcal{L}$ needs to compute the combined values with $\ell r_1 \sum_{\tau \in \mathcal{N}} r'_\tau \lambda_\tau$ exponentiations. In particular, we do not count the network time delay among the different authorities. When the attribute number is less than 4, the key generation is efficient, taking only 0.23s, 0.66s, 1.34s on average, respectively. The size of the message used in encryption and decryption phase is small (i.e., the 128_bit symmetric key). Therefore, the encryption and decryption phase are efficient with the increasing of attributes that take no more than 1s to encrypt and decrypt the message (cf. Fig. 6 (b) and (c)). The time performance is comparable with [27], which is predictable due to the similar encryption and decryption algorithm, using a constant number of pairings.



(a) Key generation phase **(b) Encryption phase** **(c) Decryption phase**

Fig. 6. Experimental results for the key generation, encryption and decryption algorithms with different attributes.

VII. Related Work

A. Vehicular Digital Forensics (VDF)

In the VDF, the investigators need to master some forensics skills that include using the software and hardware tools, even dismantling and reassembling parts of the car [7], [8], [31]. There are two important components in the vehicle, i.e., insurance black box (IBB) and Event Data Recorder (EDR) [7]. Generally, IBB is a “black box” device which records the car information, such as travelled distance, driving speed, braking, and cornering events. EDR is in charge of saving the data with regard to crash accidents. Daily *et al.* proposed a developmental model for vehicle EDR forensics [8]. They utilized the digital forensics methods to preserve the information in EDR. Nhien-An *et al.* introduced some challenges and cases on VDF based on EDR in modern vehicles [7]. Some research works focused on how to allow authorized mobile devices to connect the car. Berla iVe[9] constructs a collection of software−hardware tools to support the investigators to conduct the vehicle forensics [31]. Mansor *et al.* proposed a vehicle forensics method that stored the data in remote cloud or server, which aimed to ensure the safety of the forensics data [31], [40].

B. Blockchain Technology for Digital Forensics

A number of efforts have been made for digital forensics in recently years [7], [10], [11], [15], [16], [29], [31], [41], [42]. Among them, Mumin *et.al* proposed a blockchain based framework for forensics applications of vehicles [11], which is the most relevant with our scheme. The proposed scheme connected different stakeholders to construct a permissoned blockchain network. Once an accident happened, their scheme could reconstruct the accident scene and determine the faulty party. Compared with Mumin’s scheme, we centered on two security goals: accountability and privacy preservation. The court and law enforcement are assumed to be honest-but-curious parties, which raises additional unsolved challenges.

Decoster *et al.* designed HACIT2, a blockchain based application for dynamic navigation and forensics in VANET [43]. HACIT2 did not rely on the services of third parties while being able to ensure dynamic navigation rerouting and user anonymity. Auqib *et al.* proposed a blockchain-based digital forensics scheme named Forensic-Chain [32]. They aimed to ensure the integrity and tamper-resistant of data record by leveraging blockchain technology in digital forensics chain of custody. As for the internet of things (IoT) forensics, Jung *et al.* [44] and Duc *et al.* [45] proposed a decentralized investigation framework for digital forensics. Their schemes aim to ensure the integrity and non-repudiation of the IoT data, and enable the investigation process to be transparent. We note that most of these schemes focus on the integrity of the forensics data, while none of them consider the fine-grained data access control.

C. Accountability and Privacy in Digital Forensics

The research on *accountability* and *privacy-preserving* in digital forensics have been conducted for many years. Joshua and Dan have proposed a protocol for accountable warrant execution [15]. Their protocols could guarantee the accountability and secrecy of data records and requests. A secure IBE scheme with secret sharing of the master secret key is designed to resist SPoF/C. Jonathan *et al.* designed a practical accountable scheme for secret processes [10], allowing that the public could audit whether the surveillance powers were not abused. A set of courts exist in their scheme that they could exchange secret data, which is different from our system model. Compared with [10] and [15], the public ledger in [10] or the auditor in [15] were mainly utilized to maintain the data record, while we leverage the blockchain technology to model the VDF process as a customized state machine and achieve fine-grained access control.

VIII. Conclusion

In this paper, we analyzed the security and privacy issues of VDF in depth. Then, we proposed a blockchain-based scheme named BB-VDF to achieve accountable and secure digital forensics in the vehicular networks. Specifically, in order to achieve fine-grained access control while preserving the privacy of access structure without relying on any central party, we formalize the DKP-ABE-H scheme and give the security proof in a formal way. The whole procedures of VDF are modeled as an FSM, enabling parties to cooperate with accountability. The proposed scheme can not only mitigate

[9]https://berla.co/

malicious investigators from abusing their powers, but can eliminate the SPoF/C issue. The security of forensics data is guaranteed with the underlying blockchain. Finally, we implemented a prototype on the public Ethereum testnet *Rinkeby* to validate the feasibility and practicability.

Acknowledgments

Jian Weng was supported by National Key Research and Development Plan of China under Grant Nos. 2020YFB1005600, National Natural Science Foundation of China under Grant Nos. 61825203, U1736203 and 61732021, Major Program of Guangdong Basic and Applied Research Project under Grant No. 2019B030302008, and National Joint Engineering Research Center for Network Security Detection and Protection Technology. Ming Li was partially supported by Guangdong Basic and Applied Basic Research Foundation Grant No. 2020A1515111175, 2019B1515120010, Guangdong KeyR&D Plan2020 No. 2020B0101090002, 2020B0101090004.

References

[1] “Automotive software systems complexity: Challenges and opportunities,” ”https://slideplayer.com/slide/15897985/”, [Online].

[2] J. Wang, J. Liu, and N. Kato, “Networking and communications in autonomous driving: A survey,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1243–1274, 2018.

[3] A. V. Shvetsov, V. A. Sharov, and S. V. Shvetsova, “Method of protection of pedestrian zones against the terrorist attacks made by means of cars including off-road vehicles and trucks,” *European Journal for Security Research*, vol. 2, no. 2, pp. 119–129, 2017.

[4] S. Perry, B. Hasisi, and G. Perry, “Who is the lone terrorist? a study of vehicle-borne attackers in israel and the west bank,” *Studies in Conflict & Terrorism*, vol. 41, no. 11, pp. 899–913, 2018.

[5] “Nice attack death toll rises to 86 as injured man dies,” ”https://www.bbc.com/news/world-europe-37137816”, [Online].

[6] J. Lacroix, K. El-Khatib, and R. Akalu, “Vehicular digital forensics: What does my vehicle know about me?” in *Proceedings of the 6th ACM Symposium on Development and Analysis of Intelligent Vehicular Networks and Applications*. ACM, 2016, pp. 59–66.

[7] N.-A. Le-Khac, D. Jacobs, J. Nijhoff, K. Bertens, and K.-K. R. Choo, “Smart vehicle forensics: Challenges and case study,” *Future Generation Computer Systems*, 2018.

[8] J. S. Daily, N. Singleton, B. Downing, and G. W. Manes, “Light vehicle event data recorder forensics,” in *Advances in Computer and Information Sciences and Engineering*. Springer, 2008, pp. 172–177.

[9] A. Nieto, R. Roman, and J. Lopez, “Digital witness: Safeguarding digital evidence by using secure architectures in personal devices,” *IEEE Network*, vol. 30, no. 6, pp. 34–41, 2016.

[10] J. Frankle, S. Park, D. Shaar, S. Goldwasser, and D. Weitzner, “Practical accountability of secret processes,” in *27th {USENIX} Security Symposium ({USENIX} Security 18)*, 2018, pp. 657–674.

[11] M. Cebe, E. Erdin, K. Akkaya, H. Aksu, and S. Uluagac, “Block4forensic: An integrated lightweight blockchain framework for forensics applications of connected vehicles,” *IEEE Communications Magazine*, vol. 56, no. 10, pp. 50–57, 2018.

[12] S. Zawoad, A. K. Dutta, and R. Hasan, “Towards building forensics enabled cloud through secure logging-as-a-service,” *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 2, pp. 148–162, 2015.

[13] Y. Zhang, C. Xu, N. Cheng, H. Li, H. Yang, and X. Shen, “Chronos+: An accurate blockchain-based time-stamping scheme for cloud storage,” *IEEE Transactions on Services Computing*, vol. 13, no. 2, pp. 216–229, 2019.

[14] M. Baza, M. Nabil, M. M. E. A. Mahmoud, N. Bewermeier, K. Fidan, W. Alasmary, and M. Abdallah, “Detecting sybil attacks using proofs of work and location in vanets,” *IEEE Transactions on Dependable and Secure Computing*, 2020.

[15] J. Kroll, E. Felten, and D. Boneh, “Secure protocols for accountable warrant execution,” *See http://www. cs. princeton. edu/felten/warrant-paper. pdf*, 2014.

[16] U. Karabiyik, “Building an intelligent assistant for digital forensics,” 2015.

[17] T. V. Asaph Azaria, Ariel Ekblaw, “Medrec: Using blockchain for medical data access and permission management,” in *2nd International Conference on Open and Big Data, OBD 2016*, Vienna, Austria, Aug. 2016, pp. 25–30.

[18] D. Liu, A. Alahmadi, J. Ni, X. Lin *et al.*, “Anonymous reputation system for iiot-enabled retail marketing atop pos blockchain,” *IEEE Transactions on Industrial Informatics*, 2019.

[19] A. Kaci and A. Rachedi, “Toward a machine learning and software defined network approaches to manage miners’ reputation in blockchain,” *Journal of Network and Systems Management*, vol. 28, no. 3, pp. 478–501, 2020.

[20] M. Li, J. Weng, A. Yang, W. Lu, Y. Zhang, L. Hou, J.-N. Liu, Y. Xiang, and R. Deng, “Crowdbc: A blockchain-based decentralized framework for crowdsourcing,” *IEEE Transactions on Parallel and Distributed Systems*, 2018.

[21] Y. Lu, Q. Tang, and G. Wang, “Zebralancer: Private and anonymous crowdsourcing system atop open blockchain,” in *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2018, pp. 853–865.

[22] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2009.

[23] A. Kaci and A. Rachedi, “Poolcoin: Toward a distributed trust model for miners’ reputation management in blockchain,” in *2020 IEEE 17th Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2020, pp. 1–6.

[24] G. Wood *et al.*, “Ethereum: A secure decentralised generalised transaction ledger,” *Ethereum project yellow paper*, vol. 151, pp. 1–32, 2014.

[25] T. P. Pedersen, “Non-interactive and information-theoretic secure verifiable secret sharing,” in *Annual International Cryptology Conference*. Springer, 1991, pp. 129–140.

[26] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, “Secure distributed key generation for discrete-log based cryptosystems,” in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 1999, pp. 295–310.

[27] S. Hohenberger and B. Waters, “Attribute-based encryption with fast decryption,” in *International Workshop on Public Key Cryptography*. Springer, 2013, pp. 162–179.

[28] N. H. Ab Rahman, W. B. Glisson, Y. Yang, and K.-K. R. Choo, “Forensic-by-design framework for cyber-physical cloud systems,” *IEEE Cloud Computing*, vol. 3, no. 1, pp. 50–59, 2016.

[29] O. Henniger, “Evita: E-safety vehicle intrusion protected applications,” *tech. rep., EVITA*, 2011.

[30] J. Garay, A. Kiayias, and N. Leonardos, “The bitcoin backbone protocol: Analysis and applications,” in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2015, pp. 281–310.

[31] H. Mansor, K. Markantonakis, R. N. Akram, K. Mayes, and I. Gurulian, “Log your car: The non-invasive vehicle forensics,” in *2016 IEEE Trustcom/BigDataSE/ISPA*. IEEE, 2016, pp. 974–982.

[32] A. H. Lone and R. N. Mir, “Forensic-chain: Blockchain based digital forensics chain of custody with poc in hyperledger composer,” *Digital Investigation*, vol. 28, pp. 44–55, 2019.

[33] M. Chase, “Multi-authority attribute based encryption,” in *Theory of cryptography conference*. Springer, 2007, pp. 515–534.

[34] J. Han, W. Susilo, Y. Mu, and J. Yan, “Privacy-preserving decentralized key-policy attribute-based encryption,” *IEEE transactions on parallel and distributed systems*, vol. 23, no. 11, pp. 2150–2162, 2012.

[35] F. Zhang, E. Cecchetti, K. Croman, A. Juels, and E. Shi, “Town crier: An authenticated data feed for smart contracts,” in *Proceedings of the 2016 aCM sIGSAC conference on computer and communications security*, 2016, pp. 270–282.

[36] C. Rackoff and D. R. Simon, “Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack,” in *Annual International Cryptology Conference*. Springer, 1991, pp. 433–444.

[37] H. Yang, W. Shin, and J. Lee, “Private information retrieval for secure distributed storage systems,” *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 12, pp. 2953–2964, 2018.

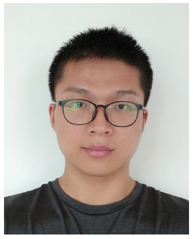
[38] S. Dziembowski, L. Eckey, and S. Faust, “Fairswap: How to fairly exchange digital goods,” in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2018, pp. 967–984.

[39] Y. Rouselakis and B. Waters, “Practical constructions and new proof methods for large universe attribute-based encryption,” in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 2013, pp. 463–474.

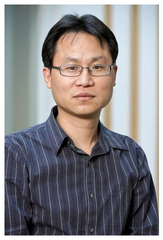
[40] A. Yang, J. Xu, J. Weng, J. Zhou, and D. S. Wong, "Lightweight and privacy-preserving delegatable proofs of storage with data dynamics in cloud storage," *IEEE Transactions on Cloud Computing*, 2018.
[41] S. Ballou, *Electronic crime scene investigation: A guide for first responders*. Diane Publishing, 2010.
[42] N. Haddadou, A. Rachedi, and Y. Ghamri-Doudane, "A job market signaling scheme for incentive and trust management in vehicular ad hoc networks," *IEEE transactions on vehicular technology*, vol. 64, no. 8, pp. 3657–3674, 2014.
[43] D. Kevin and B. David, "Hacit2: A privacy preserving, region based and blockchain application for dynamic navigation and forensics in vanet," in *International Conference on Ad Hoc Networks*. Springer, 2018, pp. 225–236.
[44] J. H. Ryu, P. K. Sharma, J. H. Jo, and J. H. Park, "A blockchain-based decentralized efficient investigation framework for iot digital forensics," *The Journal of Supercomputing*, pp. 1–16, 2019.
[45] D.-P. Le, H. Meng, L. Su, S. L. Yeo, and V. Thing, "Biff: A blockchain-based iot forensics framework with identity privacy," in *TENCON 2018-2018 IEEE Region 10 Conference*. IEEE, 2018, pp. 2372–2377.

Ming Li received his B.S. in electronic information engineering from University of South China in 2009, and M.S. in information processing from Northwestern Polytechnical University in 2012. From 2016, he started his Ph.D. at Jinan University. His research interests include blockchain, crowdsourcing, and its privacy and security.

Jian Weng is a professor and the Executive Dean with College of Information Science and Technology in Jinan University. He received B.S. degree and M.S. degree from South China University of Technology in 2001 and 2004 respectively, and Ph.D. degree at Shanghai Jiao Tong University in 2008. His research areas include public key cryptography, cloud security, blockchain, etc. He has published 80 papers in international conferences and journals such as CRYPTO, EUROCRYPT, ASIACRYPT, TCC, PKC, CT-RSA, IEEE TPAMI, IEEE TDSC, etc. He received the Young Scientists Fund of the National Natural Science Foundation of China in 2018, and the Cryptography Innovation Award from Chinese Association for Cryptologic Research (CACR) in 2015. He served as General Co-Chair for SecureComm 2016, TPC Co-Chairs for RFIDsec'13 Asia and ISPEC 2011, and program committee members for more than 40 international cryptography and information security conferences. He also serves as an associate editor of IEEE Transactions on Vehicular Technology.



Jia-Nan Liu is a Ph.D. student of Jinan University. He was born in July, 1992. He received B.S. degree and M.S. degree at Zhengzhou University and Jinan University in 2013 and 2016 respectively. His research interesting includes cryptography and cloud computing security.



Xiaodong Lin (M'09-SM'12-F'17) received the PhD degree in Information Engineering from Beijing University of Posts and Telecommunications, China, and the PhD degree (with Outstanding Achievement in Graduate Studies Award) in Electrical and Computer Engineering from the University of Waterloo, Canada. He is currently an associate professor in the School of Computer Science at the University of Guelph, Canada. His research interests include computer and network security, privacy protection, applied cryptography, computer forensics, and software security. He is a Fellow of the IEEE.

Charlie Obimbo is an associate professor at the Department of Computing and Information Science, University of Guelph. He received his Ph.D. in 2000 from the University of New Brunswick in Canada. He has worked as an assistant professor both at the University of New Brunswick, and the University of Prince Edward Island, before joining the University of Guelph in 2001. His areas of research include computer and network security, applied cryptography, and analysis and design of computer algorithms.