Towards Vehicular Digital Forensics from Decentralized Trust: An Accountable, Privacy-preservation, 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 evidences 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 decentralized solution based on blockchain for VDF named BB-VDF, in which the accountable protocols and privacy-preservation 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. Then, 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—Vehicular digital forensics, blockchain, accountability, privacy-preservation.

1 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 autonomous driving industry effectively in the near future [2].

However, as everything has its two sides, the increasing vehicles also bring 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 to a building or a crowd of people. For example, in July 14, 2016, a 20-ton rental truck rammed into the crowd who were watching a firework display in Nice, France, which has 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 while can cause catastrophic disasters to the 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 being collected by in-vehicle computers. It can help law enforcement agency 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

- *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.*

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

investigate car accident 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 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, it is not easy to conduct a vehicle forensics investigation due to the existence of several security issues. Let's take the following hypothetical scenario as an example:

Example: *Carl is a terrorist attacker. He intends to rent a car as the criminal tool. When Carl appears in a specific place, a pedestrian, Alice, discovers that Carl's behaviors are suspicious. Thus, Alice calls the policeman Bob and tells him the necessary information. Bob launches a VDF to investigate the case immediately. He first applies for a warrant from the court. After being authorized, Bob uses the permitted warrant to request the historic data on that car from related parties, such as the corresponding rental company, traffic management center, and maintenance service provider.*

Actually, there are several security and privacy issues in the above example that may have adverse impacts on the implementation of VDF: 1) First, the details of the warrant may be leaked to Carl by malicious external attackers, which makes Carl alert and changes his behavior temporarily. 2) Second, Bob may violate the purview of the warrant to require more data than being authorized. He may even tamper the collected evidences. 3) Third, there exist malicious insiders in the related parties who might modify the historical data before presenting to Bob, or claim the historical data 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. Specifically, as vehicles become smarter and more complicated than before, it is hard for the law enforcement agency to get forensics data due to the lack of specialized tools, but to appeal for technical help from a commercial party. However, it may bring the potential threat of privacy leakage. Besides, since the court has released large number of warrants accumulatively, he may forget to trace the warrants' states, which allows semi-honest investigators to still use these warrants to require sensitive data (even though they are expired) [10]. It is not an easy work for the court to trace the states of all released warrants in reality.

We note that while some existing schemes have been proposed to address parts of the issues [10]–[13], most of them are for different applications, and under different system models or security threats. Specifically, the public should be able to audit the process of VDF while preserving the privacy, which assures the accountability and legitimacy of the forensics process without powers abusing or misusing. In addition, the forensics data should be securely obtained by the law enforcement agency with fine-grained access control, nothing more and nothing less. It is non-trivial to consider the above security issues and challenges in VDF scenario simultaneously. Closest to our work is [11] which is the first research that proposes a framework on integrating different parties' data to conduct the vehicle forensics through blockchain. However, they do not resolve the challenges that 1) the confidentiality of warrants should be preserved during the forensics process (especially for the terrorist attacks), and 2) the law enforcement agency or other participants may behave dishonestly.

Therefore, we explore the potential of blockchain, and take advantage of cryptographic primitives to design a blockchain-based VDF solution with achieving accountability, privacy preservation, and fine-grained data access control. We construct a permissioned blockchain to integrate data from different data sources to assist the law enforcement agency to accomplish an investigation. The procedures of VDF is modeled as a finite-state machine (FSM) in smart contracts, which enables involved parties to cooperate and behave legally under the supervision of the public. In a nutshell, our **contributions** can be summarized as follows:

- We propose a blockchain-based VDF solution with accountability and privacy preservation named BB-VDF, in which the privacy of the warrant details and forensics data are preserved by leveraging the customized cryptographic algorithms. To generalize the forensics data access control in a fine-grained manner, a distributed KP-ABE with partially hidden access structures scheme (short for DKP-ABE-H) is designed to to protect the privacy of the warrant and eliminate single point of failure/compromise (SPoF/C) issues on secret key management.
- We design a set of smart contracts that model the vehicle forensics process as an FSM. The FSM follows the *verification-then-forwarding* pattern, in which any state transition needs to satisfy the pre-designed conditions and should be publicly verified by the majority of blockchain nodes without privacy breaches.
- 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 3, we formalize the system model, threat model, and the security goals. In section 4, we present the construction of DKP-ABE-H and analyze its security. In section 5, we present the proposed concrete scheme, including security proof and analysis. In section 6, we present the experiments and evaluation results. The related works are given in section 7. Finally, we give the conclusion in section 8.

2 Background and Preliminaries

2.1 Digital Forensics

As illustrated in Fig. 1, a typical digital forensics can be depicted as two main phases: 1) the first phase is the warrant authorization that law enforcement agency (e.g.,

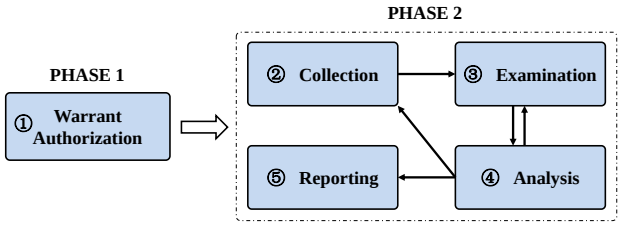


Fig. 1. The workflow of digital forensics.

the policeman) requests a valid authorization from the court before accessing data of any individual entity [14]. The warrant contains a signature from the court to permit the law enforcement agency to conduct the investigation. 2) The second phase is on data processing that contains four steps: *collection, examination, analysis*, and *reporting* [15]. Specifically, *collection* is the process of evidences 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 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.

2.2 Blockchain and Smart Contract

Most recently, the blockchain technology has been employed in many applications, such as financial services, healthcare [16], internet-of-things [17] and crowdsourcing [18], [19]. It is essentially a distributed ledger that is maintained by a number of network nodes (also called blockchain nodes) [20]. Blockchain nodes may 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 preciously, the blockchain is composed of a series of consecutive *blocks*, i.e, an ordered hash chain. Each *block* contains a number of *transactions*. Its security assurance is based on the 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: 1) *Complete Decentralization*: it is based on distributed P2P network that many untrusted nodes can achieve fair data exchange without reliance on a central party. 2) *Correct Execution*: blockchain is a global computer that each blockchain node can trace and verify the correctness of the data computation. 3) *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) [21] that facilitates the process of an application to be executed automatically and verifiably. People could participate in one DApp by providing valid inputs through on-chain transactions to call a function in smart contract.

2.3 Cryptography Algorithms

In our constructions, we make use of the following cryptographic algorithms as building blocks to achieve the 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 [22]. It allows n parties to collectively generate a key pair (i.e., public key and private key) without letting any single party to 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.* [23] improved the security of DKG protocol with the uniform randomness property. In [23], 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 Police Attribute-Based Encryption (KP-ABE): KP-ABE scheme is a type of public key encryption. It allows user 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 [24]:

-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 as input the public parameters PK, a private key SK, and a ciphertext CT associated with a set of attributes S. The algorithm will decrypt the ciphertexts and return a message M.

3 System and Security Models

In this section, we formally present the blockchain-based VDF framework and system model. Then, we give our

TABLE 1
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 $(1, ..., n)$, $(1, ..., \ell)$, respectively.
$(\mathcal{A}_1, ..., \mathcal{A}_n)$	The n decryption authorities.
$(\mathcal{S}_1, ..., \mathcal{S}_m)$	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.

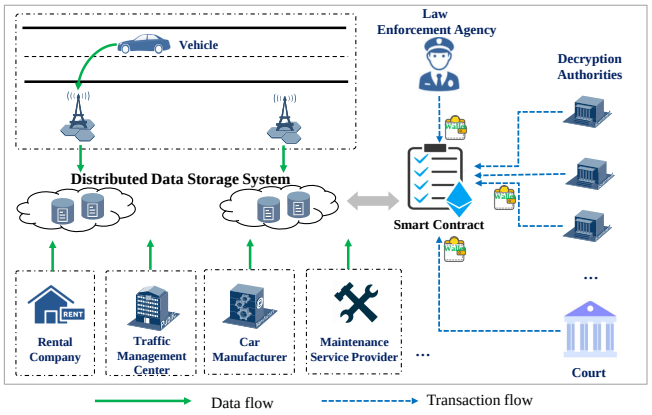


Fig. 2. System model.

security assumptions and discuss the threat model. Finally, the security goals are clearly defined. The notations that will be utilized are presented in TABLE 1.

3.1 System Architecture

The architecture of the proposed scheme is illustrated in Fig.2. It consists of six entities: Data Sources, Law Enforcement Agency, Court, Decryption Authorities, Blockchain Nodes, Data Storage Nodes. Each party has a corresponding key pair (i.e., public key and private key).

- Data Sources: identified by $\mathcal{S} = \{\mathcal{S}_1, ..., \mathcal{S}_o\}$, refer to the parties who generate the forensics data which are helpful for VDF and stores them to the data storage based on forensics-by-design paradigm [25].
- Law Enforcement Agency: identified by $\mathcal{L}$, refers to the investigator (e.g., the policeman) who is responsible for launching a digital forensics investigation. $\mathcal{L}$ has the 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 can approve $\mathcal{L}$'s request to conduct an investigation on a vehicle by following specified legal standard.
- Decryption Authorities, identified by $\mathcal{A} = \{\mathcal{A}_1, ..., \mathcal{A}_n\}$, 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 the data nodes that stores the related forensics data. Our scheme adopts the distributed data storage techniques, e.g., S3. Data are encrypted in $\mathcal{D}$ that if $\mathcal{L}$ intends to retrieve data, he should be granted with a corresponding decryption key.

The authorities initializes the system by generating the public parameters. Upon a suspicious behavior or an accident happen, $\mathcal{L}$ first applies for 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 this investigation, each procedure is finished by submitting a transaction to prompt state machine transition in smart contracts, which enables the public to audit the validity and legitimacy of the investigation. The underlying $\mathcal{B}$ is built atop of the existing blockchain architecture in our scheme, e.g., Enterprise Ethereum Blockchain. It supports Turing-complete smart contract (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, including transaction fee, we leave the incentive design to our future work.

Specifically, to obtain the forensics 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 [26] is adopted in our scheme that data request should be authorized, which ensures ECUs with secure communications for on-board system. When it is necessary to collect directly from the vehicle (e.g., traffic accidents), the forensics daemon communicates with different ECUs through Controller Area Network (CAN) bus requires access authorization, which guarantees the security of data collection.

3.2 Security Model

Here we discuss threats to digital forensics. Without loss of generality, the security of $\mathcal{B}$ follows the *majority-honest-assumption* [18], [27] that is, if most of the blockchain nodes are honest, $\mathcal{B}$ will perform the pre-defined computation correctly. The forensics daemon are assumed run securely inside OBU that malicious attackers can not disturb the normal running, or tamper the content of the collected data. Similar with the previous schemes on digital forensics, most of the entities are honest and perform their duties properly,

and all of them are assumed with bounded computation. Moreover, $\mathcal{L}$, $\mathcal{C}$, and honest $\mathcal{A}$ do not collude during the investigation. However, there still exist some malicious insiders or external attackers who may undermine the forensics investigation. Specifically, we mainly focus on the following security threats:

Threat 1: Malicious Law Enforcement Agency $\mathcal{L}$. Generally, $\mathcal{L}$ belongs to a trusted institution in reality. However, there exist some malicious insiders who are in pursuit of individual profit and do not follow the standardized digital forensics process. Thus, we consider that $\mathcal{L}$ is untrustworthy in our threat model. In particular, 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, malicious $\mathcal{L}$ (denote as $\mathcal{L}^*$) may attempt to 1) acquire more data without explicit authorization, 2) take advantage of an expired warrant to obtain access to data, or 3) 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. $\mathcal{C}$ is also 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 malicious $\mathcal{A}$ (denotes as $\mathcal{A}^*$, no more than a pre-defined ratio) might exist. They attempt to learn and exposure the details of an investigation, e.g., the identity of a suspicious vehicle. In addition, they may collude with suspicious vehicles, which allows them to change their behaviors.

4 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.

4.1 Syntax of DKP-ABE-H

Formally, the proposed decentralized key-policy attribute-based encryption with partially hidden access structures scheme (or simply DKP-ABE-H) 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 parties: the sender, the receiver, the authorities:

- *Setup*$(1^\eta, t, n)$*: is a stateful setup algorithm is 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 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 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*.*

4.2 Security Definition of DKP-ABE-H

Our security model of DKP-ABE-H is similar with the typical selective-set model as in [28], [29]. 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 attributes 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 set D, E and two integer counter $j = 0, l = 0$. 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 SubPolicyHiddenGen($\mathbb{A}$), SubKeyGen($\{\varphi_j\}_{j\in|\ell|}, \alpha_i$) and 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 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.

4.3 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 [23]. 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:

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

where λ_j is the Lagrange interpolation coefficient for sharing secret α. The authorities collaboratively specifies 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 attributes 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 [24], while he can use α_j to compute the intermediate parameters $\{g^{\mu_{1,j}}, ..., g^{\mu_{\ell,j}}\}$ as follows:

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

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:

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

$\mathcal{A}_j$ 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:

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

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:

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

$\mathsf{Decrypt}(SK, C) \rightarrow M$ 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:

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

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

4.4 Proofs of DKP-ABE-H

4.4.1 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), $\mu_{i,\tau} = W_i \cdot \overrightarrow{v_\tau} = W_i \cdot (\alpha_\tau, y_2, ..., y_n), i \in |\ell|$. $\mathcal{L}$ 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:

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

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 [24], $\mathcal{L}$ can recover the final decryption key. □

4.4.2 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.*

The proof of the Theorem 1 is shown in Appendix A.

4.4.3 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. □

5 The Proposed 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. Specially, section 5.1 outlines the overview and depicts the blockchain based forensics state machine. Section 5.2 introduces the concrete scheme and section 5.3 analyzes the security of BB-VDF.

5.1 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[3]. 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 detail 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 the 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 requires 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 state change should 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 "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

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

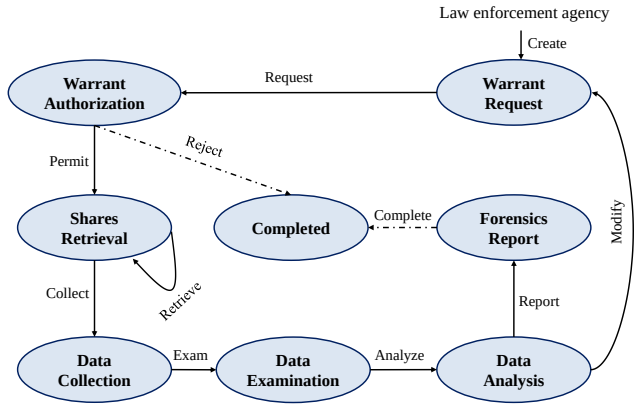


Fig. 3. The state machine model for vehicle digital forensics.

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 to have a clear understanding on the state of the forensics investigation, we leverage the off-the-shelf blockchain oracle [30] to schedule specific tasks and pass the external information to smart contracts for execution. The oracle program is initialized by $\mathcal{C}$ and 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 pre-defined time, then the corresponding entities will be informed by the blockchain oracle, or the state could be automatically transited to Completed under the setting.

5.2 Concrete Scheme

In this subsection, we present the concrete BB-VDF scheme to demonstrate the usage of DKP-ABE-H and blockchain in forensics scenario.

5.2.1 System Initialization

In BB-VDF, the court $\mathcal{C}$ and the authorities $\mathcal{A}$ are responsible for initializing the whole system Setup$(1^\eta, t, n)$ algorithm in the initialization phase. More precisely, let H_1, H_2 be two cryptographic hash functions: $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p$, $H_2 : \mathbb{G}_T \rightarrow \{0,1\}^\kappa$, 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 $MSK = \{\alpha\}$. Besides, involved parties generate their own key pairs (i.e., public key and private key). We assume that their public keys are authenticated by $\mathcal{C}$ and published in the blockchain that the public can verify the validity of a signature [4].

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

5.2.2 Digital Forensics Data Generation

During this phase, data sources are required to periodically submit the related data to $\mathcal{D}$. The period can be flexibly set, such as one hour or an half hour for the consideration of real-time forensics. A forensics 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 the simplified analysis, we assume the vehicles submit data at 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").

Specifically, 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 pieces of data are encrypted with hybrid encryption method and sent to $\mathcal{D}$. Meanwhile, $\mathcal{S}_i$ is required to submit a transaction for each data $d_{id,type,t}$, where the transaction contains the metadata (e.g., hash value, timestamp). To reduce the number of transactions, $\mathcal{S}_i$ can construct a Merkle Hash tree for the consecutive data (e.g., $\beta_{t_1 \sim t_\varsigma} = \{\beta_{t_1}, ..., \beta_{t_\varsigma}\}$) and store the Merkle root to the blockchain. Specifically, the process of the data generation can be depicted based on the Encrypt(M, PK, S) algorithm as follows:

- 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:

$$\begin{aligned} &k_1 = H_2(\varepsilon_1), \hat{C} = g^{s_1}, M_{id,type,t} = Enc_{k_1}(d_{id,type,t_1}), \\ &H_{M_{id,type,t_1}} = H_1(M_{id,type,t_1}), C = \varepsilon_1 \cdot e(g,g)^{\alpha s_1}, \\ &\{C_x = H_1(x)^{s_1}\}_{x \in \{0,1\}^*}, \theta_1 = H_1(id||type||t_1), \end{aligned} \tag{13}$$

  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 periods 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:

$$root = MerkleRoot(H_{M_{id,type_1,t_1}}, ..., H_{M_{id,type_m,t_\varsigma}}) \tag{14}$$

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

$$TX^1_{\mathcal{S}_i} = [root, (\theta_1, ..., \theta_{m \times \varsigma}), Ti]_{K^s_{\mathcal{S}_i}}, \tag{15}$$

  where Ti refers to the generation timestamp of the transaction.

5.2.3 Warrant Request

Warrant request is based on the SubPolicyHiddenGen$(\mathbb{A})$ algorithm in DKP-ABE-H. To guarantee the auditability and validation of the forensics process, the algorithm is split into two steps which are achieved as an interactive process between the agency $\mathcal{L}$ and the court $\mathcal{C}$.

Concretely, in case of accidents or suspicious behavior, $\mathcal{L}$ can require a warrant from $\mathcal{C}$ to by submitting a transaction

to create a new state machine. The warrant contains a short description des (i.e., metadata) and an access structure $\mathbb{A}$ (i.e., target identity, data type, and time). $\mathcal{L}$ sends the warrant to $\mathcal{C}$ privately through a secure channel. If $\mathcal{C}$ approves the investigation, he will sign on the warrant and generate intermediate parameters which is used to recover the decryption key. Otherwise, the investigation will be rejected and the state transits into Completed. $\mathcal{L}$ is allowed to require forensics data from multiple data sources, different data types and timestamp by constructing a comprehensive access access $\mathbb{A}$, e.g., *("*id*: David" OR "*id*: Carl") AND (2020/06/21 22:00 ≤"*time*"≤ 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.

$$TX^2_{\mathcal{L}} = [des, H_{req}, Ti]_{K^s_{\mathcal{L}}}, \tag{16}$$

  where $H_{req} = H_0(req)$ is computed as the unique identifier of the warrant in $\mathcal{B}$. The public can audit the process of a legitimate forensics investigation based on the identifier. An FSM instance is created in smart contract as: $* \xrightarrow{TX^2_{\mathcal{L}}}$ 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 Completed). Otherwise, $\mathcal{C}$ selects ℓ sets of random numbers $r_1, ..., r_\ell \leftarrow \mathbb{Z}_p$ and generates the following values:

$$\begin{aligned} &\xi_1 = (H_1(\rho(1))^{r_1}, g^{r_1}, \forall d \in \Gamma/\rho(1), H_1(d)^{r_1}), \\ &... \\ &\xi_\ell = (H_0(\rho(\ell))^{r_\ell}, g^{r_\ell}, \forall d \in \Gamma/\rho(\ell), H_0(d)^{r_\ell}), \\ &res = (\xi_1, \xi_2, ..., \xi_\ell), \end{aligned} \tag{17}$$

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

$$TX^3_{\mathcal{C}} = [H_{req}, H_1(res), H_0(\rho(1))^{r_1}, ..., H_0(\rho(\ell))^{r_\ell}, Ti]_{K^s_{\mathcal{C}}}. \tag{18}$$

Once $\mathcal{B}$ confirms the transaction $TX^3_{\mathcal{C}}$, the FSM instance will be transited into a new state as: Warrant Request $\xrightarrow{TX^3_{\mathcal{C}}}$ Warrant Authorization.

5.2.4 Auditable Data Collection

This phase mainly focuses on retrieving the decryption key from multiple authorities. Specifically, $\mathcal{L}$ computes the blinded values on res and sends it to each authority who will evaluate whether to provide the share to $\mathcal{L}$ based on their clues and judgement. If $\mathcal{L}$ receives a set $\mathcal{N}$ of $t+1$ decryption key shares from $\mathcal{A}$, he can recover the decryption key.

To protect the privacy of investigation, we design that the attributions $\rho(1), ..., \rho(\ell)$ is not revealed to any party. Therefore, $\mathcal{L}$ chooses 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, and then outputs the parameters $\{\varphi_1, ..., \varphi_\ell\}$ to each authority as in equation (4). Then, $\mathcal{L}$ can send a transaction to the blockchain to apply for the decryption key. Meanwhile, the FSM instance is transited as Warrant Authorization $\xrightarrow{TX^4_{\mathcal{L}}}$ Shares Retrieval.

$$\begin{aligned} TX^4_{\mathcal{L}} = [&H_{req}, H_0(\rho(1))^{r_1}, ..., H_0(\rho(\ell))^{r_\ell}, \\ &H_1(\varphi_1), ..., H_1(\varphi_\ell), Ti]_{K^s_{\mathcal{L}}}. \end{aligned} \tag{19}$$

After that, each authority will check whether the required decryption key is corresponding to an authorized warrant (by $TX^3_{\mathcal{C}}$ and $TX^5_{\mathcal{L}}$). If yes, he/she will join to recover the decryption key under the SubKeyGen$(\{\varphi_j\}_{j\in|\ell|}, \alpha_i)$ algorithm. Specially, to prevent $\mathcal{L}$ or the public from learning the value of $g^{\mu_{a,j}}$, $\mathcal{A}_j$ chooses a random number $r'_j \in \mathbb{Z}_p$ and computes the secret share $\widetilde{SK_j} = (\widetilde{\varphi_{1,j}}, ..., \widetilde{\varphi_{\ell,j}})$. In particular, to allow the public to verify that the provided shares are indeed computed according to the authorized warrant, we can let the authority to approve that $(\widetilde{\varphi_{1,j}}, ..., \widetilde{\varphi_{\ell,j}})$ are indeed computed based on the authorized parameters $res = (\xi_1, ..., \xi_\ell)$ using zero knowledge proof technique [31]. In addition, each authority is required to submit a transaction to prove the validity of share-providing. For instance, the transaction submitted by $\mathcal{A}_j$ is shown as follows:

$$TX^5_{\mathcal{A}_j} = [H_{req}, g^{\mu_{1,j}} H_0(\rho(1))^{r_1 r'_j}, ..., g^{\mu_{\ell,j}} H_0(\rho(\ell))^{r_\ell r'_j}, Ti]_{K^s_{\mathcal{A}_j}}. \tag{20}$$

Actually, $\mathcal{A}_j$ does not need to reveal the secret share α_j, while $\mathcal{L}$ can still figure out the blinded shares. $\mathcal{L}$ 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 FSM instance is automatically transited into the new state as: Shares Retrieval $\xrightarrow{\{TX^5_{\mathcal{A}_\tau}\}_{\tau\in\mathcal{N}}}$ Data Collection.

$\mathcal{L}$ 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 $\mathcal{D}$ [32]. 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: Data Collection $\xrightarrow{TX^6_{\mathcal{L}}}$ Data Examination by $\mathcal{L}$.

$$TX^6_{\mathcal{L}} = [H_{req}, H_1(H_{D_{id,type,t}}), Ti]_{K^s_{\mathcal{L}}} \tag{21}$$

5.2.5 Data Examination and Analysis

In this phase, $\mathcal{L}$ comprehensively conducts systematic search of the collected data, which is to identify the potential evidences. Firstly, $\mathcal{L}$ needs to prove that the integrity of the evidences should be preserved, which can be verified by using the Merkle tree proof verification algorithm [33]. Then, several kinds of software and hardware methods are utilized to extract valuable evidences [7]. After that, the state machine is transited as Data Examination $\xrightarrow{TX^7_{\mathcal{L}}}$ Data Analysis by $\mathcal{L}$. Data analysis is to further analyze the evidences

collected in data examination phase. Here, we do not depict how to use these methods to obtain the final evidences, which is out of the scope of our paper. It is worth noting that if $\mathcal{L}$ requires more data to be collected, he needs to reapply for a new warrant with following the procedures. If no, the state machine will be transited into Forensics Report (denotes as $TX^8_{\mathcal{L}}$) by $\mathcal{L}$.

5.2.6 Automated Vehicle Forensics Reporting

Based on the time-scheduling service in the blockchain oracle, our scheme allows the final vehicle forensics report to be sent different parties automatically. For instance, if a traffic accident happened and dispute existed, the final reports are required to send to drivers and insurance company simultaneously, which is to help them to make a judgement on indemnity. For the sake of the following investigation, the digest of the reports is recorded in $\mathcal{B}$ (denotes as $TX^9_{\mathcal{L}}$ by $\mathcal{L}$). After $\mathcal{C}$ and $\mathcal{L}$ confirm the final report, the state machine will be transited into Complete (denotes as $TX^{10}_{\mathcal{C}}$, $TX^{10}_{\mathcal{L}}$, respectively).

As shown in Fig. 4, we present the BB-VDF contract for the whole state machine transition, where each function is called by the corresponding party. In the "Complete" phase, when $\mathcal{L}$ accomplishes an investigation, the BB-VDF contract requires a multi-signature transaction [34] from $\mathcal{L}$ and $\mathcal{C}$ to terminate the FSM instance.

5.3 Security Analysis of BB-VDF

In this section, we discuss the security goals on accountability and privacy preservation of the proposed scheme.

5.3.1 Accountability

Complete Process Audit. As illustrated in Fig. 3, we model the process of VDF as a complete FSM instance, in which each state transition is accomplished with a valid blockchain transaction. The critical data records which are helpful for the auditability are stored in the transaction that no one can repudiate. Specifically, the verification algorithms have been pre-defined in smart contracts. As long as the underlying blockchain is secure, it is impossible for malicious attackers to disturb the normal running of smart contracts and prompt the state machine transition without being authorized and verified. Note that there are 8 states in our designed state machine and at least $(10 + t)$ transactions need be written into the blockchain (if $\mathcal{C}$ authorizes the warrant request), i.e., $TX^2_{\mathcal{L}}, TX^3_{\mathcal{C}}, TX^4_{\mathcal{L}}, \{TX^5_{\mathcal{A}_\tau}\}_{\tau\in\mathcal{N}}$ ($t + 1$ transactions), $TX^6_{\mathcal{L}}, ..., Tx^{10}_{\mathcal{C}}, TX^{10}_{\mathcal{L}}$. These data records (similar to *audit logs* in [14]) on a specific warrant are recorded in the blockchain with tamper-resistance and traceability. Thus, the BB-VDF scheme has achieved complete process audit that malicious behaviors will be detected.

Public Verifiability. As analyzed above, the whole process of any investigation is recorded in $\mathcal{B}$ which is a permissioned-based blockchain in our design, while the data in $\mathcal{B}$ are not confidential that the public can read data in each transaction. Thus, the state transitions for each warrant are transparent that any unauthorized or illegal behaviors can be detected. In addition, as long as the defined security assumption holds, $\mathcal{L}$ can not acquire more data than the actually authorized by $\mathcal{C}$. The public could verify whether

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}, π_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 = (π_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.

the shares are computed based on the authorized warrant using the intermediate parameters. And also, zero knowledge proof can be utilized to publicly verify the validity of the shares provided by $\mathcal{A}$.

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., $\left(H_0(\rho(1))^{r_1}, ..., H_0(\rho(\ell))^{r_\ell}\right)$) which will be used to recover SK. During the collection of the secret shares for recovering the decryption key, each authority will check whether the values are validated by $\mathcal{C}$ in $Tx^3_{\mathcal{C}}$. If no, the

authority refuses to provide the share.

5.3.2 Privacy Preservation

Confidentiality of Warrant. In our scheme, the secrecy of a warrant details is guaranteed against malicious or curious users. More preciously, during the process of warrant authorization, $\mathcal{L}$ sends the warrant details req to $\mathcal{C}$ through the secure channel without exposing 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, malicious users can not learn about the attributions, which is based on the intractability of discrete logarithm problem (DLP).

Confidentiality of Forensics Data. The forensics data mainly refer to the data collected by the data sources $\mathcal{S}$. According to our design, $\mathcal{L}^*$ cannot 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.

As for an honest-but-curious $\mathcal{C}$, there are two ways to obtain the decryption key: first, 1) $\mathcal{C}$ colludes with $\mathcal{L}$ who can provide 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 [23], 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, as for the public values $\{\widetilde{SK_\tau}\}_{\tau\in\mathcal{N}}$, since the random numbers $r_{\epsilon_1,d}$ are kept secret, it is impossible for malicious users to restructure the final secret shares $\{SK_\tau\}_{\tau\in\mathcal{N}}$. As for $\mathcal{A}$, it is also apparent that they can not collude to recover the decryption key under the standard *t-of-n* threshold assumptions of secure DKG.

5.3.3 Data Security

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 available of data and the available of forensics service. First, the data is stored in distributed data storage system, which enables any party who has the decryption key and download link to download the data. Second, the underlying blockchain $\mathcal{B}$ is a decentralized architecture which can resist against distributed deny of service (DDoS) attacks, and 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.

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 Merkle hash tree before an investigation, and no one can tamper these records. Thus, if $\mathcal{L}$ needs to present data evidences to the court, it is easy to prove the intergity of the data evidences using the Merkle verification algorithms. In addition, if the data are modified off chain, it can be easily detected with the hash values recorded in $\mathcal{B}$.

Unforgeability. Since each data record which will be used in the forensics and judgement should have the digital signature, it is apparent for the unforgeability. Specifically, we consider two scenarios that malicious attackers could forge evidences to disrupt the normal forensics process in BB-VDF. First, $\mathcal{L}^*$ might use the expired warrant or create a forged warrant to acquire 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 the attack being successful is negligible.

6 Implementation and Evaluation Results

In this section, we present the implementation results of the proposed scheme which focus on three aspects: 1) communication and computation costs analysis; 2) experiments for the time performance analysis off-chain; 3) experiments for the transaction time performance analysis on-chain. Specifically, BB-VDF was implemented in roughly 16.7k lines of Java, Javascript and Solidity. Our developed system could be accessed by the public web site *https://forensicschain.com*.

6.1 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, and H_1 is the hash function that uses H_0 to convert the message to hash values and then maps these values to $\mathbb{G}_1$ (similar with H_2). We set the system parameters that based on the type A prime-order bilinear groups with 160_bit $\mathbb{Z}_p$ and 512_bit $\mathbb{G}_1$. Specially, we implement the cryptographic parts off-chain using *CloudCrypto* which is written in the Java program language[5]. CloudCrypto has supported the large universe KP-ABE protocol as in [35]. We modify the source code to enable it to support the time performance analysis of our proposed KP-ABE scheme.

We implement the experiments run on a personal computer ("Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz", 4GB RAM). We design 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 stage: Setup, KeyGen, Encryption and Decryption. The Setup phase includes the initialization of the DKG and KP-ABE protocols. Specially, 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 on using symmetric encryption to encrypt and decrypt

5. https://github.com/liuweiran900217/CloudCrypto

the original data are straightforward, which will not be analyzed in the experiments.

To show the practicability of the proposed scheme, we implement our state machine contract (BB-VDF contracts) in the official Ethereum public test network *Rinkeby* [6]. There have been already more than 3,305,000 accounts and 4,663,268 blocks in Rinkeby in 1 July, 2019. 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 browser without running a full blockcahin node [7]. The transaction fee is defined as the same for different transactions. We perform 20 sets of experiments in the Rinkeby. Each experiment creates a new life cycle of an FSM instance.

6.2 Computation and Communication Cost Analysis

In this subsection, 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 attributes 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 2.

6.3 On-chain Performance Evaluation

In terms of on-chain performance, we focus on the performance analysis of the state machine in the Rinkeby for a warrant. We record the time consumption for each transaction involved in the execution of the warrant. There are 10 type of transactions involved in the proposed scheme. As shown in Fig. 5, the average confirmation time for a specific transaction is bout 32.89s. Namely, each transaction takes about 2 blocks time to be finally confirmed in the Rinkeby (a new block is generated in every 15s). Note that the confirmation time is not related with 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.

6. https://rinkeby.etherscan.io/
7. https://metamask.io/

TABLE 2
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.

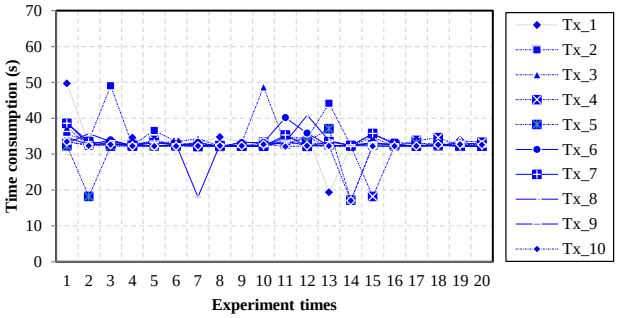
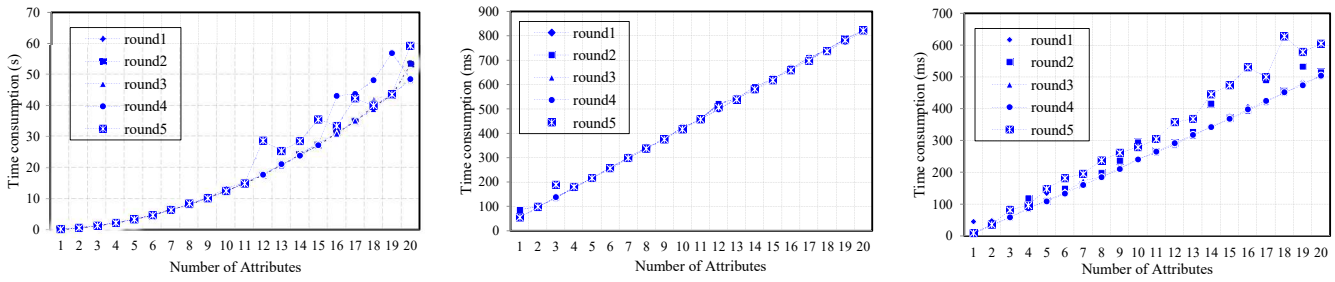


Fig. 5. The experiment results in reaching a consensus on verifying the validation of a transaction.

6.4 Off-chain Performance Evaluation

We evaluate the off-chain time performance for different phases. Specially, we conduct the experiments for 5 rounds. To test the impacts of attribute number on the computation time, we use different number of attributes in each round. We only use "AND" in the access policy. According to our 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. This is due to the fact that in order to recover SK, $\mathcal{C}$ needs to generate the parameters $\xi_1, ..., \xi_\ell$, and $\mathcal{L}$ computes the blinded values of these parameters. In addition, to recover the last secret key SK, $\mathcal{L}$ 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 consider the network time delay among the different authorities. It is worth noting that as the attributes number is less than 4 (i.e., 1, 2, and 3), the key generation is efficient that takes only 0.23s, 0.66s, 1.34s on average, respectively. As mentioned above, the size of the message used in encryption and decryption phase is small (i.e., the 128_bit symmetric key). As shown in Fig. 6 (b) and (c), the encryption and



(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.

decryption phase are efficient with the increasing of attributes that take no more than 1s to encrypt and decrypt the message.

7 Related Work

7.1 Vehicular Digital Forensics

As for the vehicle forensics, 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], [36]. There are two important components in the vehicle, i.e., insurance black box (IBB) and Event Data Recorder (EDR) [7]. Generally, IBB is a small "black box" device which records the car information, such as travelled distance, driving speed, braking, and cornering events. E-DR is in charge of saving the data with regard to crash accident. 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]. The discussed the hardware and software solutions for VDF. On the other hand, some works focused on how to allow authorized mobile devices to connect the car. Berla iVe[8] constructs a collection of software−hardware tools to support the investigators to conduct the vehicle forensics [36]. 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 [36], [37].

7.2 Blockchain Technology for Digital Forensics

Many schemes on VDF have been proposed in recently years [7], [10], [11], [14], [15], [26], [36], [38]. 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 that the process of the investigation should be accountable and preserve the privacy. The court and law enforcement are assumed to be honest-but-curious in BB-VDF, which is also the additional unsolved research problems in [11].

Decoster *et al.* designed HACIT2, a blockchain based application for dynamic navigation and forensics in VANET [39]. HACIT2 did not rely on the services of third parties while ensuring dynamic navigation rerouting and user anonymity. Auqib *et al.* proposed a blockchain-based digital forensics scheme named Forensic-Chain [40]. They aimed to ensure the integrity and tamper-resistant of data record by leveraging blockchain technology in digital forensics chain of custody. In terms of the internet of things (IoT) forensics, Jung *et al.* [41] and Duc *et al.* [42] proposed a decentralized investigation framework for digital forensics. Their scheme aimed to ensure the integrity and non-repudiation of the IoT data, and enabled the investigation process 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. Moreover, these schemes are different from ours with respect to the threat model and security goals.

7.3 Accountability and Privacy in Digital Forensics

Researches on *accountability* and *privacy-preserving* in digital forensics have been worked for many years. Joshua and Dan have proposed a protocol for accountable warrant execution [14]. Their protocols guaranteed 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] to ensure the public could audit whether the surveillance powers were not abused. The authors considered that there were multiple courts who would exchange secret data and the data were stored in the company, which was different from the system model. Compared with [10] and [14], the the public ledger in [10] or the auditor in [14] were mainly utilized to maintain the data record with tamper-proof, while we leverage the blockchain technology to model the VDF process as a customized state machine in smart contracts.

8 Conclusion

In this paper, we analyzed the potential threats on security and privacy issues on VDF. We proposed the prototype of the BB-VDF scheme on top of blockchain to achieve accountability and fine-grained access control. The whole

8. https://berla.co/

procedures of VDF are modeled as an FSM, which enables each party to cooperate honestly and accountably. We construct the DKP-ABE-H scheme to protect the privacy of the warrant and forensics data. Furthermore, BB-VDF can mitigate malicious investigators to abuse or misuse powers and eliminate SPoF/C. The integrity of the forensics data retrieval is guaranteed with the underlying blockchain. Finally, we implemented the proposed scheme on the public test network *Rinkeby* to test the feasibility and practicability.

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

[15] U. Karabiyik, "Building an intelligent assistant for digital forensics," 2015.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2009.

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

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

[23] 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.

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

[25] 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.

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

[27] 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.

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

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] R. O'Connor, "Simplicity: A new language for blockchains," in *Proceedings of the 2017 Workshop on Programming Languages and Analysis for Security*. ACM, 2017, pp. 107–120.

[35] 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.

[36] 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.

[37] 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.

[38] S. Ballou, *Electronic crime scene investigation: A guide for first responders*. Diane Publishing, 2010.

[39] 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.

[40] 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.

[41] 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.

[42] 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.

APPENDIX A
PROOF OF THE THEOREM 1

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 attributes set S^* for 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

$$h_x = \begin{cases} g^{z_x} & if\ x \in S^* \\ g^{z_x} g^{a^i} & if\ x \notin S^*, \end{cases} \quad (1)$$

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 $\mathcal{AT}$. □