A Lightweight Authentication-Driven Trusted Management Framework for IoT Collaboration

Guanjie Cheng, Yewei Wang, Shuiguang Deng, *Senior Member, IEEE*, Zhengzhe Xiang, *Member, IEEE*, Xueqiang Yan, Peng Zhao, and Schahram Dustdar, *Fellow, IEEE*

***Abstract*—The property of Internet of Things (IoT) applications is their capability to execute tasks through the collaboration of interconnected IoT objects. However, IoT collaborations face significant challenges due to security threats that undermine their reliability. An uncertified task publisher may deceive IoT devices into executing illegal tasks, while malicious attackers may intercept and modify transmitted data. Existing works on IoT trusted management issues tend to concentrate on individual aspects, such as authentication, privacy protection, and access control. However, trusted management for IoT collaboration is a multifaceted and intricate endeavor that necessitates a comprehensive approach. To fill this gap, we propose a lightweight authentication-driven trusted management framework that includes a novel authentication and key agreement scheme to guarantee the validity of task publishers, with greatly reduced overheads compared to recent works. The framework also incorporates a distributed data storage scheme and a fine-grained access control mechanism. We record the interactive messages on the blockchain to ensure behavior traceability. We evaluate the authentication scheme through comparative experiments and formal security analysis, demonstrating its efficiency and effectiveness. The experimental results of data storage and acquisition in real-world IoT environments indicate that the proposed framework is a feasible solution for reliable IoT collaboration.**

***Index Terms*—Access control, data storage, IoT collaboration, lightweight authentication, trusted management.**

I. Introduction

THE Internet of Things (IoT) has revolutionized a wide range of fields by interconnecting ubiquitous smart devices to collect data. Valuable data collected by these smart devices give birth to numerous real-life smart applications such as smart healthcare [1], smart homes [2], and smart cities [3]. Essentially, the fulfillment of IoT applications relies on the collaborative participation of numerous objects, including network operators, application servers, IoT gateways, and a vast number of IoT devices. Therefore, the reliability of IoT collaboration is critical to ensuring the effectiveness of the IoT system.

However, with the rapidly growing number of IoT devices and demand for IoT services, IoT collaboration faces enormous security threats such as identity theft, interception data tampering, denial-of-service (DoS) attacks, and fraud attacks [1]. One significant challenge is the illegal initialization of collaborative tasks, where malicious IoT task publishers (denoted by IoT managers in the following paper) may send instructions to IoT devices to perform illegal tasks. For instance, an IoT manager can send virus commands to household appliances and vehicle-mounted terminals simultaneously, which can result in stealing an individual's life data. In addition, the devices in most IoT scenarios are unsupervised, leading to vulnerability to many security issues [4]. Attackers can easily access and destroy these devices, as well as tamper with the generated data. The collected data is also vulnerable to privacy attacks and integrity compromises during transmission. To address these issues, it is critical to establish a mutual authentication and key agreement scheme to ensure legal task execution and protect confidentiality during data transmission. Furthermore, the current IoT ecosystem relies on a trusted third party (TTP) for data storage and processing [5]. However, the TTP-based data management framework suffers from inherent limitations of centralized architecture, such as lack of scalability, low bandwidth, privacy concerns, and single point of failure [4]. Lastly, there is a possibility that data users can use illegal means to gain unauthorized access to the data, such as bribing data holders or performing impersonation attacks [6], which highlights the importance of enforcing access control on data requesters.

In recent years, various schemes for IoT trusted management have been presented to address the growing security challenges [6], [7], [8], [9], [10], [11], [12], [13], [14], [15]. However, the existing literature primarily focuses on specific aspects of the trusted management issues, such as device/user authentication protocols [7], [8], [9], [10] and data access control mechanisms [11], [12], [13], [14], [15]. However, trusted management for IoT collaboration is a systemic issue due to the complex components in the IoT service process. The separate

Manuscript received 3 April 2023; revised 8 December 2023; accepted 24 December 2023. Date of publication 3 January 2024; date of current version 12 June 2024. This work was supported in part by the Key Research Project of Zhejiang Province under Grant 2022C01145, and in part by the National Science Foundation of China under Grants 62125206 and U20A20173. *(Corresponding authors: Shuiguang Deng; Peng Zhao.)*

Guanjie Cheng and Shuiguang Deng are with the First Affiliated Hospital, Zhejiang University School of Medicine, Hangzhou, Zhejiang 310003, China, and also with the College of Computer Science and Technology, Zhejiang University, Hangzhou, Zhejiang 310027, China (e-mail: guanjiech@126.com; dengsg@zju.edu.cn).

Yewei Wang is with the Department of Micro-nano Electronics, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: jakiewangxy@gmail.com).

Zhengzhe Xiang is with the School of Computer and Computing Science, Hangzhou City University School, Hangzhou, Zhejiang 310015, China (e-mail: xiangzz@zucc.edu.cn).

Xueqiang Yan is with the Wireless Technology Lab of Huawei Technologies Co., Ltd., Shanghai 201206, China (e-mail: yanxueqiang1@huawei.com).

Peng Zhao is with the First Affiliated Hospital, Zhejiang University School of Medicine, Hangzhou, Zhejiang 310003, China (e-mail: zhaop@zju.edu.cn).

Schahram Dustdar is with Distributed Systems Group, TU Wien, 1040 Vienna, Austria (e-mail: dustdar@dsg.tuwien.ac.at).

Digital Object Identifier 10.1109/TSC.2023.3349305

component solutions proposed in the state-of-the-art cannot be compatible to form a comprehensive trusted management framework. Therefore, the design of a lightweight and comprehensive trusted IoT management framework remains a pressing but unsolved challenge. To address this gap, our work commences by introducing a novel lightweight mutual authentication protocol and subsequently integrates distributed data storage and access control components to construct a comprehensive trusted framework. To the best of our knowledge, this represents the first endeavor to devise a comprehensive trusted management framework for IoT collaboration. Specifically, we propose a provably secure authentication and key agreement scheme between IoT managers and devices to prevent the launch of malicious cooperative tasks. The collected raw data is secured through symmetric encryption using the negotiated session key. Considering the resource-constrained and latency-sensitive nature of IoT devices, the authentication scheme is designed through hash and XOR operations, resulting in much lower implementation overheads compared to existing authentication schemes. Additionally, to prevent the potential security problems of the traditional centralized storage system, we introduce a distributed storage scheme based on edge computing without relying on a TTP. Finally, we incorporate a fine-grained access control mechanism to the framework to prevent unauthorized access to data.

Currently, blockchain has become one of the most appropriate candidate technologies for providing a secure and distributed ecosystem for IoT networks [16], [17], [18], [19]. The concept of a blockchain-empowered IoT has attracted considerable research interest because of the following four promising benefits: 1) it achieves decentralization and increases fault tolerance; 2) provides immutability, accountability, and traceability; 3) enhances security and confidentiality through cryptography-based consensus algorithms; 4) supports trusted cooperation by allowing automatic contract execution. In this paper, the motivation for deploying a blockchain network is to record interactive messages to ensure immutability and traceability and establish a reliable cooperation environment for IoT objects.

The main contributions are summarized as follows.

1) We propose a lightweight authentication-driven trusted management framework to ensure reliable IoT collaboration. To the best of our knowledge, this is the first attempt to design a comprehensive trusted management framework for IoT collaboration that considers mutual authentication, data confidentiality, data storage, access control, and behavior audit.
2) We propose a novel authentication and key agreement scheme that is lightweight and provably secure. This scheme ensures the legality and authenticity of IoT managers and devices and generates a negotiated key for symmetric encryption to guarantee the confidentiality of raw IoT data. The framework also includes a distributed data storage scheme and a fine-grained access control mechanism based on attribute-based encryption (ABE) to prevent unauthorized access to data.
3) We conduct comparative analysis to evaluate the computation and communication overheads of the proposed lightweight authentication scheme. The results show that the scheme is cost-efficient. We also perform a formal security analysis on the proposed authentication scheme based on the widely-used Real-Or-Random (ROR) model. Additionally, we implement a local prototype system and conduct experiments on data storage and acquisition to demonstrate the feasibility and effectiveness of the proposed trusted management framework.

The paper is organized as follows. We survey the state-of-the-art in Section II. In Section III, the system model and design goals are introduced. In Section IV, the security schemes in the proposed framework are demonstrated. In Section V, the collaboration processes and implementation are detailed and explained, respectively. The experimental results are presented and analyzed in Section VI. Section VII concludes this paper and draws future directions.

II. Related Work

With the occurrence of various security incidents, which seriously affect the quality of IoT services and system efficiency, trusted management has become a vital issue in IoT and received considerable scholarly attention in recent years. Most of the existing research on IoT trusted management is based on the hierarchical architecture of the IoT system. For example, the authors in [20] proposed a trust-based public-key management scheme for smart IoT systems based on IoT general architecture, including device layer, fog layer, and cloud layer. And the authors in [21] and [22] both utilized the three-layer system model, including data layer, core layer, and application layer to conduct the trust analysis. Differently, Wei et al. [23] proposed a three-layer research model to clarify the IoT trusted management issues from the perspective of the trust definition and the key process of trust evaluation, including data layer, computation layer, and incentive layer. Besides, the authors found that the existing research on IoT trusted issues separately focuses on the calculation of object trustworthiness, device identification, and authorization mechanisms. Therefore, the current literature on IoT trusted management can be categorized into quantification-oriented methods and system security-oriented methods. The quantification-oriented method is concerned with social trust based on objects' behaviors and service qualities [24]. On the contrary, the system security-oriented method is related to system trust based on security policies and protocols. In this paper, we concentrate on the system security-oriented direction.

Based on effective trusted management protocols, IoT collaboration can be conducted in a reliable and high-quality manner. There are two types of existing trusted management methods in the system security-oriented direction: centralized methods and decentralized methods.

Centralized trusted management methods usually require the participation of a TTP to ensure system security. The authors in [8] proposed a privacy-preserving and scalable authentication protocol for the Internet of Vehicle (IoV) with a three-layered architecture, i.e., roadside units (RSUs), RSU gateways, and a trusted authority. The authority is considered to be secure. Ding et al. [10] proposed a lightweight anonymous authentication protocol for resource-constrained IoT devices based on an information center (IC), which is assumed to be trustworthy and has powerful computing and storage resources. IC is responsible for receiving information transmitted by industrial control devices and providing related services. Zhang et al. [14] designed a secure revocable fine-grained access control and data sharing scheme for industrial Internet of Things. They used revocable identity-based proxy re-encryption in their solution, where a trusted private key generator (PKG) is needed to assign keys for devices and users. Although a third party can facilitate effective management for IoT objects, the centralized framework may suffer from scalability bottlenecks and security risks such as targeted attacks, single point of failure, DoS attack, and so on [4]. The compromise of the central party may affect the stability and scalability of the entire system.

Regarding the bottleneck issues of centralized management methods, some scholars have proposed decentralized methods to eliminate the need for a third party and enhance system scalability and security. Bera et al. [6] designed a decentralized access control protocol in IoT-enabled smart-grid system by forming the participating service providers (SP) into a peer-to-peer (P2P) blockchain network. The SPs are responsible for creating the blocks from the gathered data securely and validating the new blocks before adding them into the blockchain using the consensus algorithm. Similarly, Wang et al. [15] proposed a dynamic distributed attribute-based access control framework for blockchain-empowered IoT. The access control mechanism is designed as smart contracts that are responsible for updating attributes and trust values of IoT devices, authorization judgment, and so on. Lu et al. [25] proposed an IoT data access control scheme that combines an ABE algorithm and blockchain technology, providing fine-grained access control and efficient file sharing. However, their scheme does not account for the reliability of data sources, potentially leading to data quality or security concerns. Ahmed et al. [26] categorized current lightweight authentication methods in IoT based on different techniques and discussed the performance measures. Nevertheless, it is crucial to emphasize that trusted management in IoT extends beyond mere authentication or authorization. Vulnerabilities and compromises pervade every stage of IoT data, including collection, transmission, storage, and access. As discussed by Chanal et al. [27] in their survey, most privacy and security algorithms for IoT are still in implementation stage, reliant on several assumptions and demanding substantial resources. Existing investigations on system-oriented IoT trusted management have primarily focused on data management, identity authentication, access control, and other aspects separately, without a comprehensive framework for IoT collaboration.

In this paper, we focus on proposing a novel lightweight authentication-driven trusted management framework for IoT collaboration, that also includes a distributed data storage scheme and an access control mechanism. Authentication schemes for IoT can be categorized into three groups: certificate authority (CA)-based authentication schemes, identity-based cryptography (IBC)-based authentication schemes, and symmetric authentication schemes [4], [28]. CA-based schemes are criticized for their high certificate management costs, while IBC-based schemes are dependent on the reliability of a centralized key generation center and are vulnerable to the key escrow issue [29]. Symmetric cryptography-based schemes, on the other hand, are sensitive to the absence of non-repudiation and need to solve the key exchange problem. Moreover, these schemes may not satisfy the practical deployment requirements of IoT due to the computation complexity and resource consumption. To address this gap, we propose a novel lightweight two-factor authentication protocol that utilizes only hash and XOR operations, making it suitable for resource-constrained and delay-sensitive IoT applications. Previous endeavors have made notable contributions to IoT authentication, such as Zhou et al. [30]'s work on authentication schemes for IoT combined with cloud servers, Sharma et al. [31]'s mutual authentication scheme for cloud-IoT healthcare services, Wazid et al. [32]'s lightweight authentication and key management scheme to achieve secure communication for edge-based IoT environment, and Farash et al. [33]'s enhanced user authentication and key agreement scheme for heterogeneous wireless sensor networks. Although these works also used hash and XOR computations to enhance efficiency and practicality, they generally ignored security issues related to data storage and access, failing to deliver a comprehensive approach to IoT trusted management.

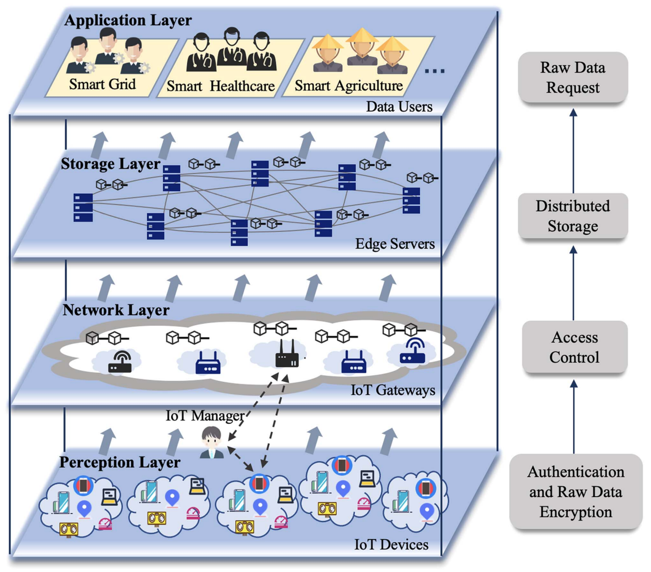


Fig. 1. Trusted IoT collaboration framework.

III. System Model and Design Goals

In this section, we illustrate the system model and discuss the security model and design goals.

A. System Model

As shown in Fig. 1, the proposed framework comprises four layers. At the bottom is the perception layer, comprising a range of smart devices, such as sensors, inductors, and wearables, that collect valuable real-time data from various sources. The IoT managers execute complex applications by launching IoT tasks that require collaboration among IoT devices from multiple fields, and as such, it is essential to mitigate the risk of potential malicious attacks by restricting their commands to IoT devices. The second layer comprises IoT gateways, which serve as the management and control unit closest to the terminal device and preprocess the data collected by the perception layer. The third layer consists of edge servers deployed in close proximity to mobile users, responsible for data storage and providing agile response services to data users. The top layer comprises data users from diverse application fields who further process and model the IoT data for specific use cases.

To ensure the legitimacy of IoT tasks, we propose a mutual authentication and key agreement protocol between IoT managers and devices. Smart devices encrypt the raw data using the negotiated session key to ensure data confidentiality. IoT gateways use ABE technology to set specific access policies that grant relevant users access to data. Edge servers cooperate with each other to form a distributed data storage system. Additionally, all gateways and edge servers maintain a blockchain network where interactive messages are stored and smart contracts are deployed. Data users interact with the nearest edge server covering them to download their requested data.

Overall, the trusted management framework has five types of entities: IoT managers, IoT devices, IoT gateways, edge servers, and data users. IoT managers and devices are unreliable since attackers may impersonate a legitimate IoT manager to launch IoT tasks. IoT devices may be compromised by malicious IoT managers and attempt to execute harmful tasks. IoT gateways are assumed to be fully trusted and perform their duies honestly and reliably. Edge servers are semi-trused (i.e., honest but curious). Although each edge server is curious about the IoT data, they still faithfully execute the proposed schemes, cannot deliberately modifying or leaking their stored data. Data users might also engage in unethical conduct, such as collaborating with other unscrupulous parties or masquerading as authorized users to gain access to desired information. Moreover, they could initiate malicious actions once they have secured access authorization.

B. Threat Model and Security Requirements

1) Threat Model: To identify potential security concerns and mitigate trust risks in the IoT collaboration system, it is important to analyze the threat model. To accomplish this, we utilize the widely-used STRIDE model [34], as detailed below.

- Spoofing: Spoofing refers to the ability to impersonate legal entity in the system. In the IoT collaboration system, the data users may masquerade as authenticated nodes to gain access illegally.
- Tampering: Tampering refers to the malicious modification of data, which violates the integrity. Attackers or malicious data holders can tamper with the collected IoT data and disrupt IoT collaboration.
- Repudiation: Repudiation indicates the ability of an attacker to make it impossible to link an action one performed to himself.
- Information Disclosure: Information disclosure threats describe the exposure of information to unauthorized individuals, which violate the confidentiality. The data transmitted in the open network are susceptible to privacy violations.
- DoS: DoS attacks deny service to valid users. Adversaries may invalidate IoT data to violate service availability by launching a DoS attack.
- Elevation of Privilege: In this threat, an attacker gains privileged access without authorization and thereby can penetrate the IoT system.

Additionally, as the lightweight authentication and key agreement scheme constitutes the core novelty of the proposed trusted management framework, we use the widely-accepted ROR model et al. [35] to describe the threat model used to verify its security. We summarize a few primitives associated with the ROR model as follows.

Participants: We denote $\Pi^{n_1}_{M_i}$, $\Pi^{n_2}_{D_j}$, and $\Pi^{n_3}_{GTW}$ as the n_1th, n_2th, and n_3th instance of the IoT manager M_i, the terminal device D_j, and the gateway node GTW, respectively. These instances are also termed as $oracles$.

Partnering: The notation of partnering is defined based on session identifications (sid) and partner identification of a manager M_i or a terminal device D_j. In our system, the partner of an instance $\Pi^{n_1}_{M_i}$ of the manager M_i is the instance $\Pi^{n_2}_{D_j}$ of the terminal device D_j, and vice versa. Note that only when the following two conditions are met can $\Pi^{n_1}_{M_i}$ and $\Pi^{n_2}_{D_j}$ be called partners: 1) both instances are accepted, 2) both instances have been authenticated.

Freshness: The instances $\Pi^{n_1}_{M_i}$ and $\Pi^{n_2}_{D_j}$ are called fresh if the adversary $\mathcal{A}$ does not get the session key SK through the $Reveal$ query (defined below) to these instances or its partner.

Adversary: We assume an adversary $\mathcal{A}$ who has the full control over the wireless communication between IoT managers, terminal devices, and IoT gateways. During the communication of normal entities, $\mathcal{A}$ not only can intercept and eavesdrop but also can modify, create a new one, or directly forward the message to an intended entity. Besides, the adversary is granted the following queries:

- $Execute(\Pi^{n_1}, \Pi^{n_2})$: This query models a passive attack where the adversary eavesdrops the honest executions between a manager instance Π^{n_1} and a device instance Π^{n_2}. $\mathcal{A}$ can intercept all messages during the communication through this query.
- $Send(\Pi^{n_1}, \mathcal{M})$: This query models an active attack where $\mathcal{A}$ sends a message $\mathcal{M}$ to a participant instance Π^{n_1} and then get a response message corresponding to the transmitted $\mathcal{M}$.
- $CorruptSC(\Pi^{n_1}_{M_i})$: This query models a smart card lost attack. The output of this query consists of the information stored in the smart card SC_i of the authorized manager M_i. This query assures the weak-corruption model in which the internal data and temporary keys of the participant instances are not corrupted.
- $Test(\Pi^{n_1})$: This query captures the semantic security of SK_{ij} between the manager M_i and the device D_j. Let b be a bit set uniformly at random at the beginning of the experiment, which is kept secret from the adversary $\mathcal{A}$ and used to decide the output of $Test()$. After executing this

query, if the session key SK_{ij} has been established and is $fresh$, the instance $Test(\Pi^{n_1})$ returns SK_{ij} if $b = 1$ or a random number of the same size if $b = 0$. Otherwise, this query returns the invalid symbol $\perp$.

In addition, the participants and adversary can access a hash function $h(\cdot)$, which is modelled by a random oracle, say $Hash$. When one makes a hash query $h(x)$, $Hash$ will return the result if the same query was submitted. Otherwise, $Hash$ will return a random response chosen uniformly from its output domain and then stores the pair permanently.

In the ROR model, $\mathcal{A}$ is given access to the above oracles in an experiment and outputs a guess bit b'. If $b' = b$, $\mathcal{A}$ is said to win the experiment challenging the semantic security. Note that $\mathcal{A}$ can ask $Test(\Pi^{n_1})$ to either the manager instance or the device instance for many times. Let $Succ$ denote the event in which $\mathcal{A}$ wins this game. The advantage of $\mathcal{A}$ in violating the semantic security of the protocol $\mathcal{P}$ in the ROR sense is defined as $Adv_{\mathcal{P}}^{ake} = |2Pr[Succ] - 1|$. Note that the advantage of the adversary that simply guesses b is 0. If $Adv_{\mathcal{P}}^{ake}$ is negligible, the protocol $\mathcal{P}$ can be considered as a secure authentication and key agreement protocol.

2) Security Requirements: In addition to preventing the threat model, the proposed trusted management framework must also satisfy the following security requirements (or called design goals.)

- *Lightweight Authentication:* It guarantees that a requested IoT collaboration task is generated by a legal IoT manager and the device is also valid. Additionally, the designed authentication scheme must be lightweight to satisfy the real-time and resource-sensitive demands of IoT applications.
- *Data Confidentiality:* Any untrusted entity cannot extract private IoT data via analyzing the intercepted messages during the communication and storage.
- *Distributed Storage:* The storage of IoT data requires a distributed, consistent, and secure system that does not rely on a third party. Moreover, the system should ensure data availability.
- *Access Control:* Only valid users authorized by data owners can obtain access rights.
- *Traceability:* The behaviors of malicious and compromised entities are capable of being traced.

Furthermore, the proposed framework should prioritize user experience without compromising it for the sake of improving security.

IV. Proposed Lightweight Authentication-Driven Trusted Management Framework

In this section, we present a detailed description of the components in our trusted management framework, including a novel lightweight authentication and key agreement scheme, a distributed data storage scheme, and an access control mechanism.

A. Authentication and Key Agreement Scheme

As mentioned earlier, a valid but malicious IoT manager could induce terminal devices to perform illegal tasks, causing serious system threats. We propose an efficient and provably secure authentication and key agreement protocol to avoid such a trouble. Considering the expensive distribution and maintenance costs of the traditional certificate-based authentication schemes, we intend to design a lightweight protocol that can adapt to resource-constrained IoT terminal devices by adopting simple hash functions and XOR operations. We assume IoT devices communicate with each other through Low Power Wide Area (LPWA) methods, such as NB-IoT, while data users communicate with IoT devices and edge servers through cellular networks.

TABLE I
Notation Description

Notation	Description
K_{GTW}	The secret of GTW
ID_i and DID_j	The identity of M_i and D_j
Tag_i and Tag_j	The secret label of M_i and D_j
PW_i	The password of M_i
T	Timestamp
SC_i	The smart card of M_i
K_i and K_j	Session key computation materials
$SKey$	The negotiated session key
SK	The user secret key
$h(\cdot)$	Cryptographic one-way hash function
a||b	Concatenation of a and b
$SymEnc$	Symmetric encryption algorithm
$SymDec$	Symmetric decryption algorithm

Table I explains the primary notations used in the following paper. The detailed processes are illustrated in Fig. 2. We assume the gateway is equipped with a long-lived secret K_{GTW}. The target terminal device D_j is initially allocated with a unique identity DID_j. The gateway calculates $Tag_j = h(DID_j \| K_{GTW})$ as an authenticated tag for D_j. Then DID_j and Tag_j are stored in the trusted module of D_j. The manager M_i is fitted with an identity ID_i and a password PW_i. Before sending instructions to the target device, M_i is expected to register on the network with the help of GTW. First, M_i generates a random nonce r_i and calculates $W_i = h(r_i \| PW_i)$. Then he sends $m_{reg} = (ID_i, W_i, T_{reg})$ to GTW through blockchain, where T_{reg} denotes the current timestamp. Blockchain ensures the immutability and traceability of the exchanged messages. Upon receiving m_{reg}, GTW verifies ID_i and then calculates an authenticated tag $Tag_i = h(ID_i \| K_{GTW})$ and a number $\alpha_i = W_i \oplus Tag_i$ to hide Tag_i. Next, GTW creates a smart card SC_i, stores α_i in it and then issues SC_i to M_i. M_i writes r_i into SC_i after receiving it.

If M_i wants to activate D_j, he inserts SC_i into D_j, and inputs his identity ID_i and password PW_i. SC_i computes $W_i = h(r_i \| PW_i)$, $Tag_i = W_i \oplus \alpha_i$, and $\beta_i = h(Tag_i \| T_1)$. Whereafter, SC_i selects a random number K_i for subsequent key agreement. Then SC_i calculates $\Theta_i = K_i \oplus \beta_i$ and $E_i = h(\beta_i \| ID_i \| DID_j)$ to hide K_i and β_i, respectively. Afterward, SC_i sends the login request $m_1 = (ID_i, E_i, \Theta_i, T_1)$ to the terminal device D_j through blockchain. After receiving m_1, D_j first check whether $|T_1 - T_c| \leq \Delta T$, where T_c is the current timestamp and ΔT is the accepted transmission delay. If the verification fails, D_j rejects the request.

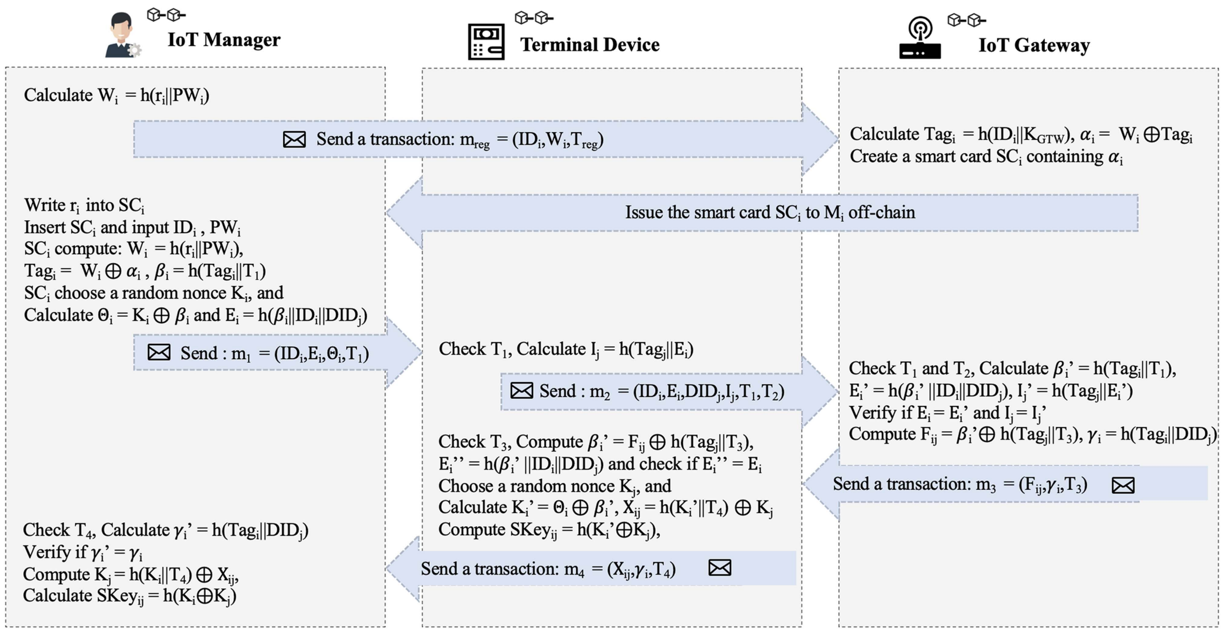

IoT Manager
Terminal Device
IoT Gateway
Calculate W_i = h(r_i||PW_i)
Send a transaction: m_reg = (ID_i,W_i,T_reg)
Calculate Tag_i = h(ID_i||K_GTW), α_i = W_i ⊕ Tag_i
Create a smart card SC_i containing α_i
Issue the smart card SC_i to M_i off-chain
Write r_i into SC_i
Insert SC_i and input ID_i , PW_i
SC_i compute: W_i = h(r_i||PW_i),
Tag_i = W_i ⊕ α_i , β_i = h(Tag_i||T_1)
SC_i choose a random nonce K_i, and
Calculate Θ_i = K_i ⊕ β_i and E_i = h(β_i||ID_i||DID_j)
Send : m_1 = (ID_i,E_i,Θ_i,T_1)
Check T_1, Calculate I_j = h(Tag_j||E_i)
Send : m_2 = (ID_i,E_i,DID_j,I_j,T_1,T_2)
Check T_1 and T_2, Calculate β_i' = h(Tag_i||T_1),
E_i' = h(β_i' ||ID_i||DID_j), I_j' = h(Tag_j||E_i')
Verify if E_i = E_i' and I_j = I_j'
Compute F_ij = β_i' ⊕ h(Tag_j||T_3), γ_i = h(Tag_i||DID_j)
Send a transaction: m_3 = (F_ij,γ_i,T_3)
Check T_3, Compute β_i' = F_ij ⊕ h(Tag_j||T_3),
E_i'' = h(β_i' ||ID_i||DID_j) and check if E_i'' = E_i
Choose a random nonce K_j, and
Calculate K_i' = Θ_i ⊕ β_i', X_ij = h(K_i'||T_4) ⊕ K_j
Compute SKey_ij = h(K_i'⊕K_j),
Send a transaction: m_4 = (X_ij,γ_i,T_4)
Check T_4, Calculate γ_i' = h(Tag_i||DID_j)
Verify if γ_i' = γ_i
Compute K_j = h(K_i||T_4) ⊕ X_ij,
Calculate SKey_ij = h(K_i⊕K_j)

Fig. 2. Authentication and key agreement protocol.

Otherwise, D_j computes $I_j = h(Tag_j \| E_i)$. Then D_j sends $m_2 = (ID_i, E_i, DID_j, I_j, T_1, T_2)$ to GTW. Similarly, GTW checks whether T_1 and T_2 are valid to decide whether to continue or terminate. Afterward, GTW calculates $\beta_i' = h(Tag_i \| T_1)$, $E_i' = h(\beta_i' \| ID_i \| DID_j)$, and $I_j' = h(Tag_j \| E_i')$. D_j verifies whether $E_i' = E_i$ and $I_j' = I_j$. If the verification succeeds, GTW accepts M_i since Tag_i is valid. Then GTW computes $F_{ij} = \beta_i' \oplus h(Tag_j \| T_3)$ and $\gamma_i = h(Tag_i \| DID_j)$ as authentication materials for D_j and M_i, respectively. Lastly, it replies $m_3 = (F_{ij}, \gamma_i, T_3)$ to D_j. D_j checks T_3 in the same way aforementioned once receiving m_3. Then it computes $\beta_i' = F_{ij} \oplus h(Tag_j \| T_3)$ and $E_i'' = h(\beta_i' \| ID_i \| DID_j)$. If $E_i'' = E_i$, it represents the manager authentication is successful. Otherwise, the manager's request is declined. Subsequently, D_j randomly selects a secret number K_j for session key computation. D_j obtains $K_i' = \Theta_i \oplus \beta_i'$ and calculates $X_{ij} = h(K_i' \| T_4) \oplus K_j$ to hide K_j. Then the session key $SKey_{ij}$ can be computed as $SKey_{ij} = h(K_i' \oplus K_j)$. Finally, D_j packs $m_4 = (X_{ij}, \gamma_i, T_4)$ as a transaction and sent to M_i through blockchai for key agreement. Once receiving m_4, M_i first examines T_4 and then calculates $\gamma_i' = h(Tag_i \| DID_j)$. If γ_i' is equal to γ_i, M_j can conclude that the sender is legitimate. Afterward, M_i computes $K_j = h(K_i \| T_4) \oplus X_{ij}$. Finally, M_i can get the session key $SKey_{ij}$ calculated as $SKey_{ij} = h(K_i \oplus K_j)$. Therefore, $SKey_{ij}$ can be used to safeguard the communications between M_i and D_j.

It should be emphasized that longer secret keys can significantly enhance the security of XOR operations, thus reinforcing the security of the proposed authentication scheme. Additionally, to further strengthen the system's security, it is advisable for IoT devices and managers to update their authenticated tags periodically.

B. Distributed Data Storage Scheme

To mitigate the trust issues that arise from centralized storage systems, including single point attacks and privacy breaches, we propose a decentralized storage environment using edge computing. We use the Advanced Encryption Standard (AES) for raw data encryption with the key negotiated during the authentication phase, denoted as $SKey$. The encrypted data, represented as $Edata$, are then stored on distributed edge servers based on the Kademlia algorithm. This algorithm is widely used in Peer-to-Peer (P2P) networks for data storage and retrieval due to its simplicity, flexibility, and security [36]. Each node in the Kademlia network is assigned a 160-bit identity (ID) as a unique identifier, while the hash value of the encrypted IoT data block serves as a 160-bit *Key*, and the encrypted data block as *Value*. When a new server joins the Kademlia network, it is assigned a randomly generated 160-bit node ID. The *Key-Value* pair is then stored on several servers whose ID values are close to the *Key*. The maximum number of nodes that Kademlia can accommodate is 2^{160}, which greatly exceeds the number of servers in the actual network.

Each server in the storage system maintains a portion of the storage contents, rather than the complete ledger. Additionally, the state information of relevant servers is stored in each node by a K-bucket mechanism. The Kademlia algorithm uses XOR to calculate the distance between nodes. For instance, the distance between the edge server with ID 01000000 and another server with ID 00000001 is calculated as follows:

$$01000000 \oplus 00000001 = 01000001. \tag{1}$$

As shown in Fig. 3, each edge server maintains a 160-layer K-bucket table, with each K-bucket storing state information

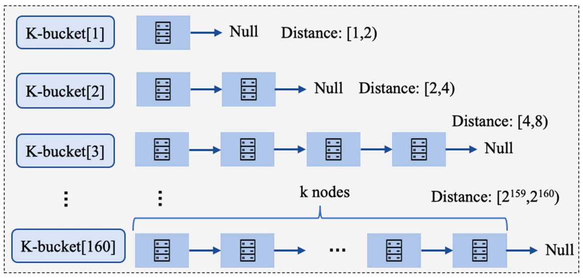


Fig. 3. Storage structure of edge servers.

of k nodes within the range of the interval [2^{i-1}, 2^i]. The state information includes the node's ID, Internet Protocol (IP) address, and access port. k is a system-level constant, set by the storage system itself. With this mechanism in place, a server can locate a target server's information with at most $\log_2 n$ queries, where n is the total number of edge servers in the system.

The proposed distributed storage scheme, based on the Kademlia algorithm, effectively addresses two critical issues that traditional storage systems encounter. First, frequent node entries and exits in a distributed system result in heavy network traffic and reduced storage and search efficiency. To alleviate these issues, our system utilizes a partitioned storage approach, with each node maintaining messages of parts of servers instead of complete state information of other nodes. This minimizes the impact on the network when a participant changes its state. Second, traditional distributed storage frameworks are vulnerable to global privacy leakage in case of malicious server behavior. In contrast, our storage scheme provides enhanced data security by limiting each node's state information to parts of the servers. These features significantly improve service experience while ensuring data security.

C. Access Control Mechanism

In a trusted IoT system, it is expected that data can only be accessed by an authorized user. To meet this demand, we put forward an ABE-based access control mechanism in the trusted management framework. This mechanism enables data owner to specify data access policies based on user identities and attributes, thus achieving fine-grained access management over data. Users can download an encrypted data block from the edge server, but the raw data cannot be decrypted without proper authorization. The core of the access control mechanism is the authorization of $SKey$. Most ABE-based access control schemes require a trusted attribute authority (AA) to set up the system and distribute corresponding secret key for data users [37]. However, the trusted authority-based system is vulnerable to key abuse, as AA can decrypt all encrypted data in the edge server by obtaining the encryption key $SKey$, resulting in serious data leakage. To mitigate this issue, we propose delegating the AA duties to a gateway with relatively strong computing power, assuming gateways are reliable. The access control mechanism is detailed below.

1) $Setup(1^\lambda) \rightarrow (PK, MSK)$: The gateway that acts as AA performs this algorithm during system initialization. It takes as input security parameter λ and outputs the system public key PK and system master key MSK. The gateway publishes PK in the system through blockchain and stores MSK locally.
2) $Encrypt(PK, P, SKey) \rightarrow Ciphertext$: This algorithm is executed by the gateway. It takes as inputs the public key PK, the access policy P, and the negotiated session key $SKey$. Notably, P is established during system initialization. Upon completion, the algorithm outputs $Ciphertext$, which is subsequently stored locally within the gateway.
3) $KeyGen(MSK, S) \rightarrow SK$: The $KeyGen$ algorithm is also executed by the gateway as well. Upon receiving a user's request for a secret key, the gateway first verifies his identity. If successfully authenticated, the user is assigned the corresponding attributes set S. The algorithm accepts the system master key MSK and the user's attribute set S as inputs and generates the secret key SK as output. Subsequently, the gateway securely stores $\{PK, SK\}$ in a private channel shared with the data user on the blockchain. Note that SK is a long-term key that will not be reclaimed until the data user's access right is revoked.
4) $Decrypt(PK, SK,$redCiphertext$) \rightarrow SKey$: The $Decrypt$ algorithm is carried out by the data user. It takes as inputs the public key PK, the user secret key SK, and $Ciphertext$. Upon execution, the algorithm outputs the encryption key $SKey$. If the user's attributes set S meets the access policy P, the data user can successfully decrypt $Ciphertext$ to recovery $SKey$, subsequently enabling the execution of the final algorithm as described below. Conversely, if the access policy is not met, the decryption will fail.
5) $SymDec(Edata)_{SKey} \rightarrow RawData$: The data user performs this symmetric decryption algorithm. $Edata$ is downloaded from the edge servers network using the content identifier (CID), which is given by the gateway after data request. This algorithm produces the required raw data as output.

Given the possibility that data users may initiate malicious actions after obtaining access permissions (i.e., acquiring SK), it is imperative to establish a revocation mechanism for access rights. A straightforward and efficient approach involves the gateway maintaining a confidential table containing authorized users' identities. Upon detecting a malicious user, the gateway removes the relevant identity from the table and documents the nefarious action. As a result, subsequent data requests from the user will be rejected, preventing the download of $Edata$ from the storage network and effectively revoking their access privileges.

V. Collaboration Processes and Smart Contracts Implementation

This section gives a brief demonstration of the processes of the proposed trusted management framework. Then, the smart contract-based implementation is described in detail.

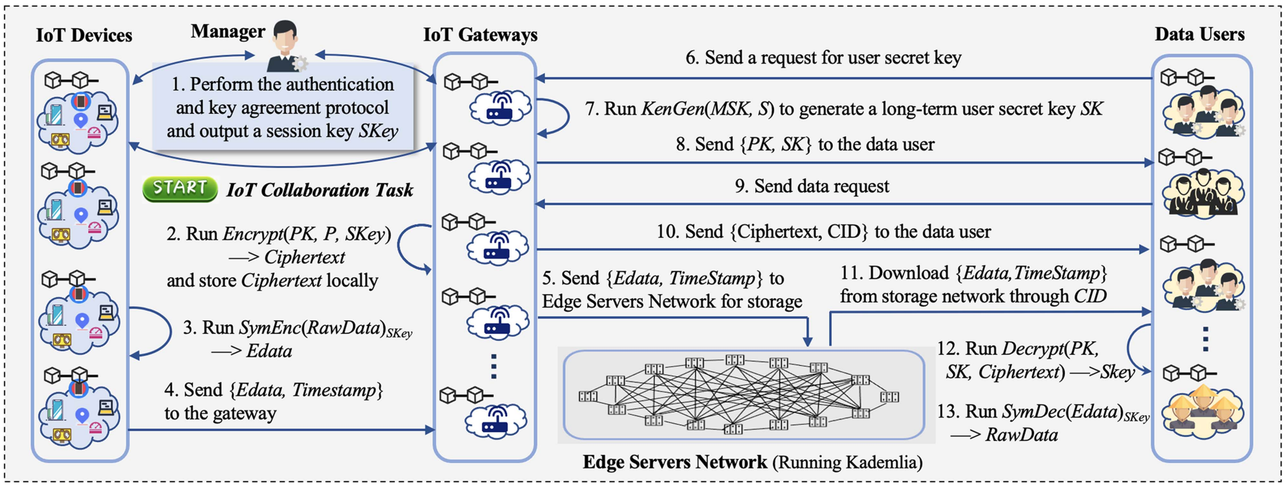

IoT Devices
Manager
IoT Gateways
Data Users
1. Perform the authentication and key agreement protocol and output a session key SKey
START IoT Collaboration Task
2. Run Encrypt(PK, P, SKey) —> Ciphertext and store Ciphertext locally
3. Run SymEnc(RawData)SKey —> Edata
4. Send {Edata, Timestamp} to the gateway
5. Send {Edata, TimeStamp} to Edge Servers Network for storage
6. Send a request for user secret key
7. Run KenGen(MSK, S) to generate a long-term user secret key SK
8. Send {PK, SK} to the data user
9. Send data request
10. Send {Ciphertext, CID} to the data user
11. Download {Edata,TimeStamp} from storage network through CID
12. Run Decrypt(PK, SK, Ciphertext) —>Skey
13. Run SymDec(Edata)SKey —> RawData
Edge Servers Network (Running Kademlia)

Fig. 4. Overall processes of the proposed trusted management framework.

A. Overall Processes

The comprehensive workflow is illustrated in Fig. 4, with some negligible steps omitted for simplicity. The entire process can be divided into three parts. $Step$ 1 executes a lightweight authentication and key agreement protocol, ensuring the reliability of both the manager initiating the collaborative tasks and the IoT device. Upon successful identity authentication, the manager commences collaboration by sending instructions to the relevant terminal device. $Steps$ $2-5$ outline the encryption and storage procedures. The session key $SKey$, negotiated during the authentication phase, is encrypted using ABE by the gateway for subsequent data access control. The ciphertext is locally stored within the gateway. Subsequently, the devices encrypt the collected raw data using a symmetric encryption algorithm with $SKey$ before sending it to the gateway. The edge servers, upon receiving $\{Edata, Timestamp\}$ from the gateway, generate and publish a hash-based identifier CID, enabling users to locate the required data within the storage system. $Steps$ $6-13$ depict the data request and acquisition processes. When the gateway receives a secret key request from a data user, it executes the $KeyGen$ algorithm to generate SK and sends $\{PK, SK\}$ to the user via a private secure channel. Note that the request events of data users are recorded on the blockchain for subsequent behavior auditing. The user then sends a data request to the gateway to obtain $\{Ciphertext, CID\}$. Following this, the user downloads $\{Edata, TimeStamp\}$ from the storage system, using $TimeStamp$ to verify data freshness. Finally, the user executes the $Decrypt$ function to obtain $SKey$ from $Ciphertext$ and performs the symmetric decryption function to get the requested raw data. Please note that the cleaning, formatting, and preprocessing of IoT data fall outside the scope of this paper.

B. Smart Contracts Implementation

With smart contracts, we provide a system where stakeholders' interaction rules can be coded as programs and then enforced autonomously without a centralized trust. We divide the whole contract implementation into two main modules: Data Storage Contract and Server Management Contract. In the contracts, we utilize the special variable *msg.sender* which always exists in the global namespace and represents the address of the contract creator or the contract caller.

Algorithm 1: InfoStorage.

```
Input  : $Edata, TimeStamp$
Output: null
1  if msg.sender is not dataOwner then
2      throw
3  end
4  datablock $\Leftarrow$ $(Edata, TimeStamp)$
5  dataID = Hash(datablock)
6  for $i = 1$ to $k$ do
7      Find Node whose ID is close to dataID
8      if Ping Node = true then
9          Store dataID and datablock
10     end
11     dataID = dataID + 1
12 end
```

1) Data Storage Contract: This contract aims to implement the storage operations and is created by the IoT gateway.

1) *Initialization:* This procedure initializes several contract variables. The *dataOwner* variable is of type *address* and represents the address of the IoT gateway. The *authorizedServers* variable is of type *mapping* and represents a collection that maps authorized edge server addresses to a bool value. To manage the edge servers, the gateway can use the relevant function interfaces of the contract to add, modify, or delete entries in the *authorizedServers* collection.
2) *infoStorage:* This contract is called by the edge servers to store $\{Edata, Timestamp\}$ into the distributed storage networks.

2) Server Management Contract: In this contract, we consider the exit and entry of edge servers in the data system. The

Algorithm 2: AddServer or RemoveServer.

```
Input  : newServerAddress
Output: bool
if msg.sender is not dataOwner then
    throw
end
addServer:
if newServerAddress exist then
    Return false
end
else
    authorizedServers[newServerAddress] ⇐ true
    Return true
end
removeServer:
if newServerAddress not exist then
    Return false
end
else
    authorizedServers[newServerAddress] ⇐ false
    Return true
end
```

Algorithm 3: DataSearch.

```
Input  : CID
Output: $Edata, Timestamp$
if msg.sender is not dataOwner then
    throw
end
for $i = 1$ to $k$ do
    Find Node with a content identifier equal to CID
    if Ping Node = true then
        Return $Edata, TimeStamp$
        break
    end
end
```

Server Management Contract is deployed by IoT gateways and generally provides the following function interfaces.

1) *addServer or removeServer Contract:* This function can only be executed by the contract's creator. The new edge server sends the identity certificate to the data owner as a registration request. Upon successful authentication, the edge server is authorized through this function. On the contrary, to remove a server from the authorized set, the data owner can provide the address of the server to be removed as an input parameter to this function
2) *dataSearch Contract:* This function is invoked by edge servers. It aims to find the corresponding $\{Edata, Timestamp\}$ in the distributed storage system using CID for the data user.

VI. Performance Analysis

We construct a real IoT environment to evaluate the performance of our proposed trusted IoT collaboration framework. First, we analyze the computational and communication overheads of the proposed authentication protocol by comparing it with four recent works [30], [31], [32], [33]. Subsequently, we investigate the efficiency of data storage utilizing the prototype. Third, we assess user experience by analyzing the execution time of ABE algorithms and the average time for data users to access their requested data across various data sizes under different access policies. Moreover, we conduct formal security verification of the authentication protocol using the widely-accepted ROR model. Finally, we discuss the STRIDE model and security requirements for the proposed trusted management framework.

A. Experimental Setup

1) Environmental Setup: We employ Raspberry Pi 4 B with Cortex-A72 (1.5 GHz) and 4 GB of memory as the IoT gateway and pyWiFi-ESP32 with 8 MB SRAM as the terminal device. For data user operations, a computer running macOS Big Sur with an Intel Core i7 processor (2.3 GHz) and 16 GB of memory is utilized. IPFS, a well-known distributed file storage and sharing system based on Kademlia, is employed as the underlying storage network. An edge server, powered by a computer running macOS Sonoma with a 10-core Apple M1 Pro chip and 16 GB of RAM, acts as an IPFS node. Additionally, Hyperledger Fabric 2.4 is deployed as the blockchain environment to facilitate event storage and secure interactions. We implement the ABE algorithm based on FAME, as proposed in [38]. For the $SymEnc$ and $SymDec$ functions, we use AES in CBC mode with 32-Byte keys. The implementation codes are accessible at https://github.com/CommuSA/TSC-2023.

2) Parameter Settings: For theoretical analysis, we set the size of an identity, a random number, a timestamp, and the output of a hash operation as 20 bytes, 4 bytes, 4 bytes, and 32 bytes, respectively. We set the data access policy of type ($Attribute_1$ **AND** $Attribute_2$ **...AND** $Attribute_N$) as in [38], where $Attribute$ is represented by ($Value_1$ **OR** $Value_2$).

B. Computation and Communication Costs of the Proposed Authentication Scheme

In this section, we evaluate the computational and communication cost of the proposed authentication protocol by comparing it with four related two-factor authentication schemes that have been reviewed in Section II, namely [30], [31], [32], [33]. We select these schemes as benchmarks due to their relevance and practical applicability. First, they closely align with the IoT architectures in our scheme, where the authors investigate mutual authentication between users (or servers) and devices with the involvement of gateways (or edge nodes). Second, these schemes all focus on two-factor authentication and key agreement. Lastly, they all demonstrate the efficiency and practicality of their schemes for real-world IoT devices, rendering them robust comparative benchmarks for our protocol.

Note that in the implementation of crypto modules, the execution time of XOR operations can be neglected when compared to the one-way hash function and the symmetric encryption/decryption algorithm. Thus, the processing time involved with XOR operations is not included. We use T_h and T_{XOR}

TABLE II
COMPARISON OF COMPUTATION COSTS (MS)

	Zhou's [30]	**Sharma's [31]**	**Wazid's [32]**	**Farash's [33]**	**Ours**
User	$12T_{XOR} + 10T_h \approx 1.20$	$6T_{XOR} + 11T_h \approx 1.32$	$5T_{XOR} + 9T_h \approx 1.08$	$6T_{XOR} + 11T_h \approx 1.32$	$4T_{XOR} + 6T_h \approx 0.72$
Device	$9T_{XOR} + 19T_h \approx 22.80$	$4T_{XOR} + 5T_h \approx 6.00$	$5T_{XOR} + 8T_h \approx 9.60$	$4T_{XOR} + 7T_h \approx 8.40$	$4T_{XOR} + 5T_h \approx 6.00$
Gateway	$16T_{XOR} + 7T_h \approx 4.90$	$6T_{XOR} + 7T_h \approx 4.90$	$8T_{XOR} + 15T_h \approx 10.50$	$6T_{XOR} + 14T_h \approx 9.80$	$T_{XOR} + 5T_h \approx 3.50$
Total Cost	28.90 ms	12.22 ms	21.18 ms	19.52 ms	10.22 ms

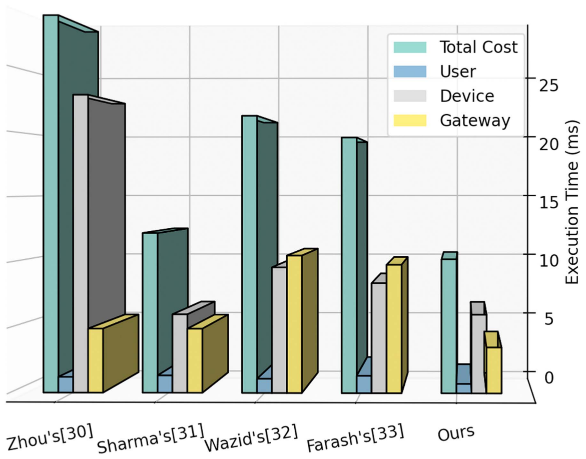


Fig. 5. Comparison results of execution time (ms).

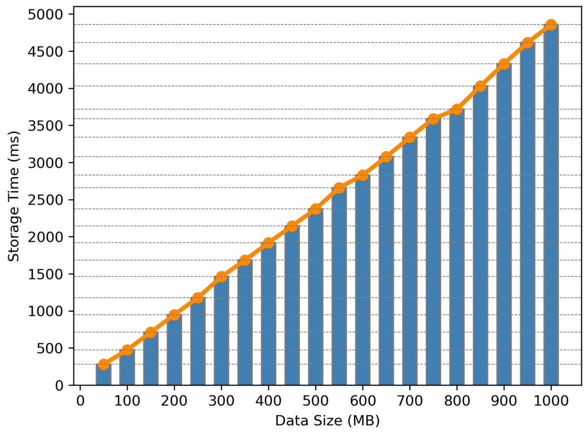


Fig. 6. Data storage time (ms).

TABLE III
COMPARISON OF COMMUNICATION COSTS (BYTES)

	Zhou's	**Sharma's**	**Wazid's**	**Farash's**	**Ours**
User	148	120	88	100	88
Device	192	292	256	304	180
Gateway	392	132	100	132	68
Total Cost	732	544	444	536	336

to denote the execution time of a general hash operation and an XOR operation, respectively. It takes 1.2 ms, 0.7 ms, and 0.12 ms to execute a single SHA256 operation with 10 KB inputs by the terminal device, gateway, and data user, respectively. Then we calculate the overall computation costs, as shown in Table II. Note that the authentication phase starts after the smart card is sent to the IoT manager. The table shows that our protocol consumes fewer resources than the other works. Fig. 5 demonstrates the execution time of our protocol and the others. The results indicate that the proposed protocol exhibits greater computational efficiency across user, device, and gateway sides, making it well-suited for resource-sensitive IoT applications.

We further evaluate the communication overheads of the proposed authentication scheme through comparative analysis with other schemes. Note that these lightweight schemes all require four message transfers for authentication. The comparative results are shown in Table III. Our findings indicate that the proposed scheme outperforms the other schemes in terms of communication costs, particularly from the perspective of the device and gateway sides (cost 180 bytes and 68 bytes respectively). Therefore, the proposed lightweight authentication scheme satisfies the low-latency requirement of IoT objects and thus well-suited for practical IoT applications.

C. *Storage Performance*

To better understand the storage performance of our proposed trusted framework, we conduct an evaluation on the Kademlia-enabled storage system. As our prototype is built on top of IPFS, we evaluate its efficiency by measuring storage time using the widely-used KDD Cup 1999 Data set. In IPFS's implementation of the Kademlia algorithm, the value of K is set to 20, which indicates that each node maintains a routing table that contains information about the 20 closest nodes in the network, based on their distance metric (XOR metric) from the node's own ID. We record the average execution results over 100 experimental runs, which are presented in Fig. 6. Our results show that the storage time increases linearly with the size of data. Specifically, it takes about 0.48 seconds and 4.86 seconds to store 100 MB and 1000 MB of data, respectively, in our prototype system. These findings demonstrate the efficacy of our proposed trusted management framework for efficient and distributed data storage in real-world scenarios.

Numerous optimization schemes for the Kademlia algorithm have been proposed in recent years, which could be utilized to enhance the storage performance of our trusted management framework. For instance, Monteiro et al. [39] suggested enhancing Kademlia performance by modifying Kademlia identifiers through the addition of prefixes. Marandi et al. [40] proposed a novel Kademlia scheme based on random linear network coding (RLNC). The simulation results demonstrated that their scheme outperforms traditional Kademlia in terms of core performance

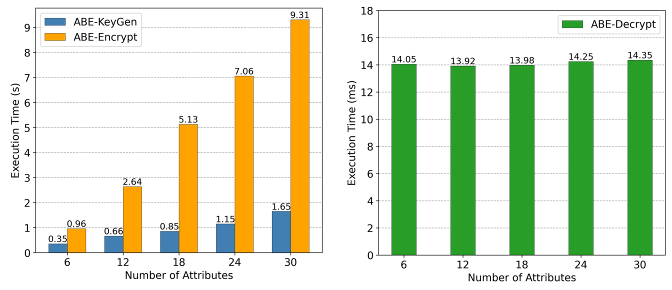


Fig. 7. Execution time of ABE algorithms.

metrics. These schemes can be integrated into our framework to further improve its storage performance.

D. User Experience

User experience is evaluated by the average time required to obtain the requested data. We establish multiple access policies to simulate access control mechanisms with varying levels of flexibility by altering the number of attributes. Initially, we observe the execution performance of the primary ABE algorithms under various access policies, with the data size initialized to 45 MB. Next, we analyze the user experience of the proposed trusted management framework by recording the data acquisition time for different numbers of attributes and various data sizes. We execute each function or operation 100 times and compute the average execution time.

Fig. 7 demonstrates the execution times of the $KeyGen$, $Encrypt$, and $Decrypt$ algorithms in ABE under different access policies. It is noteworthy that the gateway executes the $KeyGen$ and $Encrypt$ algorithms while the data user undertakes the $Decrypt$ algorithm. We find that both the execution time of encryption operations and key generation operations exhibit an increase with the number of attributes, albeit the former incurring a higher computing cost. Fortunately, the $Encrypt$ algorithm necessitates a singular execution for each session key $SKey$, contributing to efficiency gains. The execution of a $KeyGen$ operation, when the access policy involves 30 attributes, takes 1.65 seconds, affirming the practicability of our framework within real-world IoT environments. Furthermore, it is observed that decryption operations exhibit negligible sensitivity to the varying number of attributes (take about 14 milliseconds), requiring substantially less time compared to $KeyGen$ and $Decrypt$ algorithms, thereby presenting user-friendliness.

As shown in Fig. 4, $steps$ $11-13$ dominates the time required to obtain the requested data. Fig. 8 presents the time needed for data users to access data of various sizes under different access policies. The execution times for data downloading, ABE decryption, and symmetric decryption operations are documented separately. As observed from Fig. 8(a)–(c), data downloading from the distributed storage system constitutes the largest portion of the data acquisition time, while ABE decryption requires the least amount of time. Furthermore, the data size does not affect ABE decryption operations, but an increase in the number of attributes leads to a longer decryption time. This is because the $Decrypt$ function is not related to the size of the requested data. In contrast, the number of attributes does not impact data downloading or AES decryption time, both of which are affected by the data size.

Fig. 8(d) illustrates the total data acquisition time under various access policies. For example, under a policy with 12 attributes, a user takes 1.77 seconds to obtain the requested 45 MB of data in this prototype system. Clearly, in a practical environment with more powerful configurations, the execution of cryptographic operations would be faster, and parallel processing of data requests would be possible. This would ensure that the proposed trusted management framework meets the actual user experience requirements, such as low latency, reliability, and availability.

E. Security Analysis

The rigorous security analysis of the proposed framework is discussed in this section. First, we verify the security of the proposed authentication protocol based on the broadly-used ROR model [41]. Then, we discuss how the proposed trusted management framework is capable of defending against the threat model and meets the security requirements.

Theorem 1: Let $\mathbb{O}$ be a uniformly distributed dictionary from which the passwords are drawn. An adversary $\mathcal{A}$ is trying to violate the semantic security of our protocol $\mathcal{P}$ by asking the aforementioned queries in polynomial time t. Assume that it is of extreme difficulty to compromise terminal devices before establishing trust. Thus, we have

$$Adv_{\mathcal{P}}^{ake}(\mathcal{A}) \leq \frac{N_h^2}{|Hash|} + \frac{2N_{Send}}{|\mathbb{O}|}, \tag{2}$$

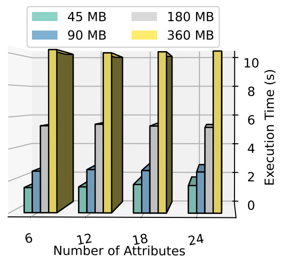
where $N_h, N_{Send}, |Hash|$ and $|\mathbb{O}|$ indicate the number of $Hash$ oracles, the number of $Send$ queries, the valid output domain of the hash algorithm, and the size of $\mathbb{O}$, respectively.

Proof: Four games are defined to prove this theorem. Let $Succ_i$ $(i = 0, 1, 2, 3)$ denote the situation of wining the game G_i where $\mathcal{A}$ manages to guess the value of b. The corresponding advantage of $\mathcal{A}$ in G_i is defined as $Pr[Succ_i]$. The detailed demonstration of each game is given below.

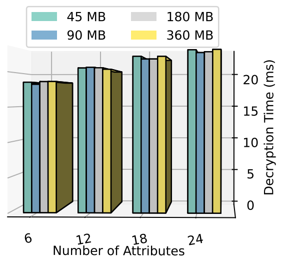
Game G_0 : In G_0, the adversary $\mathcal{A}$ performs a real attack in random oracle model against $\mathcal{P}$ and chooses b randomly at the start of the game. According to the definition, it follows that

$$Adv_{\mathcal{P}}^{ake}(\mathcal{A}) = |2Pr[Succ_0] - 1| \tag{3}$$

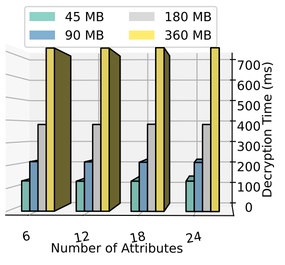
Game G_1 : In G_1, $\mathcal{A}$ asks $Execute(\Pi^{n_1}, \Pi^{n_2})$ queries to launch eavesdropping attacks and queries $Test$ oracle in the last. $\mathcal{A}$ needs to judge whether the output session key of $Test$ oracle is real or random. As described in Section III, the session key SK_{ij} is calculated from K_i and K_j. And $K_i = \Theta_i \oplus \beta_i$, $\beta_i = h(Tag_i\|T_1)$ or $\beta_i = F_{ij} \oplus h(Tag_j\|T_3)$, where $Tag_i = h(ID_i\|K_{GTW})$ or $Tag_i = \alpha_i \oplus W_i$, and $W_i = h(ID_i\|PW_i)$. We can draw a conclusion that $\mathcal{A}$ can not obtain K_i from the eavesdropped information, since α_i is stored in the smart card and PW_i and K_{GTW} are kept secretly in the device side and gateway side, respectively. Besides, since $K_j = X_{ij} \oplus h(K_i\|T_4)$, $\mathcal{A}$ must get K_i to compute K_j. Hence, $Execute$ oracle can not bring additional advantage for winning



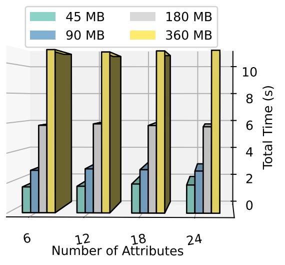
(a) Data downloading time.



(b) ABE decryption time.



(c) AES decryption time.



(d) Total execution time.

Fig. 8. Time overhead of fetching data of various sizes under different access policies.

this experiment. Thus, we have

$$Pr[Succ_1] = Pr[Succ_0] \tag{4}$$

Game G_2 : In G_2, $\mathcal{A}$ can query $Send(\Pi^{n_1}, \mathcal{M})$ and $Hash$ oracles to conduct active attacks. $\mathcal{A}$ tries to convince an entity to accept the fabricated messages. Frequent hash queries can be asked by $\mathcal{A}$ to find collisions. However, there is no collision while making $Send$ queries because each transaction is related to the identity of a participant and the timestamp, as shown in Fig. 2. In accordance with the birthday paradox, we have:

$$|Pr[Succ_1] - Pr[Succ_2]| \leq \frac{N_h^2}{2|Hash|} \tag{5}$$

Game G_3 : In this game, we add the $CorruptSC$ oracle and simulate the smart card lost attack. Using this query, $\mathcal{A}$ can obtain α_i and r_i from SC_i. Only by successfully guessing the password PW_i from $\mathbb{O}$ can $\mathcal{A}$ calculate Tag_i and eventually get K_i. The number of incorrect password inputs are always limited by the system. Hence, we have

$$|Pr[Succ_2] - Pr[Succ_3]| \leq \frac{N_{Send}}{|\mathbb{O}|} \tag{6}$$

With all the oracles been modelled in the last game G_3, the adversary has no other beneficial methods to win the experiment breaking the security of $\mathcal{P}$ instead of simply guessing the value of b. Therefore, we can infer that $Pr[Succ_3] = 1/2$. According to (2)–(5), we can deduce

$$\begin{aligned} Adv_{\mathcal{P}}^{ake}(\mathcal{A}) &= |2Pr[Succ_1] - 1| \\ &\leq |\frac{N_h^2}{|Hash|} + 2Pr[Succ_2] - 1| \\ &\leq |\frac{N_h^2}{|Hash|} + 2\left(\frac{N_{Send}}{|\mathbb{O}|}\right) + 2Pr[Succ_3] - 1| \\ &\leq \frac{N_h^2}{|Hash|} + \frac{2N_{Send}}{|\mathbb{O}|} \end{aligned} \tag{7}$$

Consequently, the proposed authentication and agreement protocol $\mathcal{P}$ is secure as long as the range of the hash function and the size of the password dictionary are large enough. Any adversary has no advantage to infer the negotiated session key. □

We discuss the threat model as follows.

- Spoofing: Even if the adversary obtains $Edata$, they cannot decrypt it to get $RawData$. Only users with attribute sets that match the access policy will be assigned the correct secret key and execute the $Decrypt$ function. Therefore, the proposed framework is effective in defending against spoofing.
- Tampering: Since $RawData$ is encrypted prior to storage, attackers have no opportunity to tamper with it. Moreover, edge servers also cannot modify the stored data since they are semi-trusted.
- Repudiation: Since the interactive messages are recorded on the blockchain, behaviors can be traced from the ledger to ensure non-repudiation.
- Information Disclosure: Attackers can not deduce any private information from $Edata$. Thus, the framework can resist information disclosure.
- DoS: The distributed architecture can efficiently disperse the traffic of DoS attacks. The proposed redundant and scalable storage mechanism can also help protect against DoS attacks.
- Elevation of Privilege: There is no possibility for a user whose attributes do not match the access policy set by the data owner to gain system access.

We discuss the security achievements as follows.

- Lightweight Authentication: Our proposed authentication scheme demonstrates a lightweight property, as confirmed by a theoretical analysis of computation and communication costs when compared to related works [30], [31], [32], [33]. The scheme guarantees that the IoT collaboration task is initiated by legal managers and executed by valid devices.
- Data Confidentiality: The collected $RawData$ is encrypted by IoT devices using a symmetric encryption algorithm $SymEnc(RawData)_{SKey} \rightarrow Edata$ before being stored. This ensures the protection of data confidentiality, making it impossible for both outside attackers and inside adversaries to access any private information from $Edata$.
- Distributed Storage: A distributed storage network for $\{Edata, Timestamp\}$ is constructed based on edge computing using Kademlia, which eliminates the reliance on any centralized party. The security, flexibility, and redundancy features of Kademlia ensure the feasibility and data availability of our storage scheme. Furthermore, the experimental results on data storage shown in Section VI-C

demonstrate the efficiency and practicality of our proposed approach.

- Access Control: The proposed ABE-based access control mechanism is implemented based on FAME [38], which is proven fully secure under the decisional linear assumption (DLIN). Only users with attributes that satisfy the policy can access the data, thus achieving fine-grained data access control for IoT collaboration. The experimental evaluation on user experiences demonstrates the efficacy of the proposed access control mechanism.
- Traceability: The proposed framework provides behaviors traceability by recording interactive messages on the blockchain. During the authentication phase, the messages $\{m_{reg}, m_1, m_2, m_3, m_4\}$ are transmitted to the target receivers via the blockchain. If dishonest IoT managers or terminal devices are suspected, supervisors can query their transactions from the blockchain. Additionally, the requests of data users are also stored on the blockchain and serve as event proof for subsequent behavior audit. Thus, the proposed trusted management framework guarantees traceability.

VII. Conclusion and Future Work

In this paper, we propose a lightweight authentication-driven trusted management framework to ensure reliable IoT collaboration. A novel lightweight authentication and key agreement protocol is designed to ensure the legality and authenticity of IoT managers and devices, and output a negotiated key to guarantee the confidentiality of the exchanged data. The framework also includes a distributed data storage scheme based on edge computing and a fine-grained access control mechanism based on ABE. The comparative analysis with related works indicates that the proposed lightweight authentication scheme is resource-efficient. Additionally, we construct a real-world IoT environment for further performance assessment. The experimental evaluation on data storage and user experience demonstrates the feasibility and effectiveness of the proposed trusted management framework.

References

[1] M. Masud, G. S. Gaba, K. Choudhary, M. S. Hossain, M. F. Alhamid, and G. Muhammad, “Lightweight and anonymity-preserving user authentication scheme for IoT-based healthcare,” *IEEE Internet Things J.*, vol. 9, no. 4, pp. 2649–2656, Feb. 2022.

[2] H. Luo, C. Wang, H. Luo, F. Zhang, F. Lin, and G. Xu, “G2F: A secure user authentication for rapid smart home IoT management,” *IEEE Internet Things J.*, vol. 8, no. 13, pp. 10 884–10 895, Jul. 2021.

[3] H. Ding et al., “Probabilistic data prefetching for data transportation in smart cities,” *IEEE Internet Things J.*, vol. 9, no. 3, pp. 1655–1666, Feb. 2022.

[4] G. Cheng, Y. Chen, S. Deng, H. Gao, and J. Yin, “A blockchain-based mutual authentication scheme for collaborative edge computing,” *IEEE Trans. Comput. Social Syst.*, vol. 9, no. 1, pp. 146–158, Feb. 2022.

[5] J. Zhang, T. Li, Z. Ying, and J. Ma, “Trust-based secure multi-cloud collaboration framework in cloud-fog-assisted IoT,” *IEEE Trans. Cloud Comput.*, vol. 11, no. 2, pp. 1546–1561, Second Quarter 2023.

[6] B. Bera, S. Saha, A. K. Das, and A. V. Vasilakos, “Designing blockchain-based access control protocol in IoT-enabled smart-grid system,” *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5744–5761, Apr. 2021.

[7] A. Vangala, A. K. Sutrala, A. K. Das, and M. Jo, “Smart contract-based blockchain-envisioned authentication scheme for smart farming,” *IEEE Internet Things J.*, vol. 8, no. 13, pp. 10 792–10 806, Jul. 2021.

[8] M. N. Aman, U. Javaid, and B. Sikdar, “A privacy-preserving and scalable authentication protocol for the Internet of Vehicles,” *IEEE Internet Things J.*, vol. 8, no. 2, pp. 1123–1139, Jan. 2021.

[9] L. Zhang, Y. Ye, and Y. Mu, “Multiauthority access control with anonymous authentication for personal health record,” *IEEE Internet Things J.*, vol. 8, no. 1, pp. 156–167, Jan. 2021.

[10] X. Ding, X. Wang, Y. Xie, and F. Li, “A lightweight anonymous authentication protocol for resource-constrained devices in Internet of Things,” *IEEE Internet Things J.*, vol. 9, no. 3, pp. 1818–1829, Feb. 2022.

[11] W. Yang, Z. Guan, L. Wu, X. Du, and M. Guizani, “Secure data access control with fair accountability in smart grid data sharing: An edge blockchain approach,” *IEEE Internet Things J.*, vol. 8, no. 10, pp. 8632–8643, May 2021.

[12] A. Saini, Q. Zhu, N. Singh, Y. Xiang, L. Gao, and Y. Zhang, “A smart-contract-based access control framework for cloud smart healthcare system,” *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5914–5925, Apr. 2021.

[13] A. Cullen, P. Ferraro, W. Sanders, L. Vigneri, and R. Shorten, “Access control for distributed ledgers in the Internet of Things: A networking approach,” *IEEE Internet Things J.*, vol. 9, no. 3, pp. 2277–2292, Feb. 2022.

[14] W. Zhang, H. Zhang, L. Fang, Z. Liu, and C. Ge, “A secure revocable fine-grained access control and data sharing scheme for SCADA in IIoT systems,” *IEEE Internet Things J.*, vol. 9, no. 3, pp. 1976–1984, Feb. 2022.

[15] P. Wang, N. Xu, H. Zhang, W. Sun, and A. Benslimane, “Dynamic access control and trust management for blockchain empowered IoT,” *IEEE Internet Things J.*, vol. 9, no. 15, pp. 12997–13009, Aug. 2022.

[16] H.-N. Dai, Z. Zheng, and Y. Zhang, “Blockchain for Internet of Things: A survey,” *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8076–8094, Oct. 2019.

[17] M. A. Ferrag and L. Shu, “The performance evaluation of blockchain-based security and privacy systems for the Internet of Things: A tutorial,” *IEEE Internet Things J.*, vol. 8, no. 24, pp. 17 236–17 260, Dec. 2021.

[18] M. A. Ferrag and L. Maglaras, “DeepCoin: A novel deep learning and blockchain-based energy exchange framework for smart grids,” *IEEE Trans. Eng. Manag.*, vol. 67, no. 4, pp. 1285–1297, Nov. 2020.

[19] M. A. Ferrag, M. Derdour, M. Mukherjee, A. Derhab, L. Maglaras, and H. Janicke, “Blockchain technologies for the Internet of Things: Research issues and challenges,” *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2188–2204, Apr. 2019.

[20] M. S. Haghighi, M. Ebrahimi, S. Garg, and A. Jolfaei, “Intelligent trust-based public-key management for IoT by linking edge devices in a fog architecture,” *IEEE Internet Things J.*, vol. 8, no. 16, pp. 12 716–12 723, Aug. 2021.

[21] C. Zhang, W. Li, Y. Luo, and Y. Hu, “AIT: An AI-enabled trust management system for vehicular networks using blockchain technology,” *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3157–3169, Mar. 2021.

[22] I. U. Din, A. Bano, K. A. Awan, A. Almogren, A. Altameem, and M. Guizani, “LightTrust: Lightweight trust management for edge devices in industrial Internet of Things,” *IEEE Internet Things J.*, vol. 10, no. 4, pp. 2776–2783, Feb. 2023.

[23] L. Wei, Y. Yang, J. Wu, C. Long, and B. Li, “Trust management for Internet of Things: A comprehensive study,” *IEEE Internet Things J.*, vol. 9, no. 10, pp. 7664–7679, May 2022.

[24] W. Z. Khan, Q.-U.-A. Arshad, S. Hakak, M. K. Khan, and S. Saeed-Ur-Rehman, “Trust management in social Internet of Things: Architectures, recent advancements, and future challenges,” *IEEE Internet Things J.*, vol. 8, no. 10, pp. 7768–7788, May 2021.

[25] X. Lu, S. Fu, C. Jiang, and P. Lio, “A fine-grained IoT data access control scheme combining attribute-based encryption and blockchain,” *Secur. Commun. Netw.*, vol. 2021, pp. 1–13, 2021.

[26] W. K. Ahmed and R. S. Mohammed, “Lightweight authentication methods in IoT: Survey,” in *Proc. Int. Conf. Comput. Sci. Softw. Eng.*, 2022, pp. 241–246.

[27] P. M. Chanal and M. S. Kakkasageri, “Security and privacy in IoT: A survey,” *Wirel. Pers. Commun.*, vol. 115, no. 2, pp. 1667–1693, 2020.

[28] X. Yang, X. Yi, S. Nepal, I. Khalil, X. Huang, and J. Shen, “Efficient and anonymous authentication for healthcare service with cloud based WBANs,” *IEEE Trans. Serv. Comput.*, vol. 15, no. 5, pp. 2728–2741, Sep./Oct. 2022.

[29] H. Sikarwar and D. Das, “Towards lightweight authentication and batch verification scheme in IoV,” *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 5, pp. 3244–3256, Sep./Oct. 2022.

[30] L. Zhou, X. Li, K.-H. Yeh, C. Su, and W. Chiu, “Lightweight IoT-based authentication scheme in cloud computing circumstance,” *Future Gener. Comput. Syst.*, vol. 91, pp. 244–251, 2019.

[31] G. Sharma and S. Kalra, "A lightweight user authentication scheme for cloud-IoT based healthcare services," *Iranian J. Sci. Technol., Trans. Elect. Eng.*, vol. 43, no. 1, pp. 619–636, 2019.
[32] M. Wazid, A. K. Das, S. Shetty, J. JPC Rodrigues, and Y. Park, "LDAKM-EIoT: Lightweight device authentication and key management mechanism for edge-based IoT deployment," *Sensors*, vol. 19, no. 24, 2019, Art. no. 5539.
[33] M. S. Farash, M. Turkanović, S. Kumari, and M. Hölbl, "An efficient user authentication and key agreement scheme for heterogeneous wireless sensor network tailored for the Internet of Things environment," *Ad Hoc Netw.*, vol. 36, pp. 152–176, 2016.
[34] K. Peng, M. Li, H. Huang, C. Wang, S. Wan, and K.-K. R. Choo, "Security challenges and opportunities for smart contracts in Internet of Things: A survey," *IEEE Internet Things J.*, vol. 8, no. 15, pp. 12 004–12 020, Aug. 2021.
[35] M. Abdalla, P.-A. Fouque, and D. Pointcheval, "Password-based authenticated key exchange in the three-party setting," in *Proc. Int. Workshop Public Key Cryptography*, Springer, 2005, pp. 65–84.
[36] P. Maymounkov and D. Mazieres, "Kademlia: A peer-to-peer information system based on the XOR metric," in *Proc. Int. Workshop Peer-to-Peer Syst.*, Springer, 2002, pp. 53–65.
[37] S. Qi, Y. Lu, W. Wei, and X. Chen, "Efficient data access control with fine-grained data protection in cloud-assisted IIoT," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2886–2899, Feb. 2021.
[38] S. Agrawal and M. Chase, "FAME: Fast attribute-based message encryption," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 665–682.
[39] J. Monteiro, P. A. Costa, J. Leitao, A. De la Rocha, and Y. Psaras, "Enriching kademlia by partitioning," in *Proc. IEEE 42nd Int. Conf. Distrib. Comput. Syst. Workshops*, 2022, pp. 33–38.
[40] A. Marandi, H. Sehat, D. E. Lucani, S. Mousavifar, and R. H. Jacobsen, "Network coding-based data storage and retrieval for kademlia," in *Proc. IEEE 93rd Veh. Technol. Conf.*, 2021, pp. 1–7.
[41] J. Zhang, H. Zhong, J. Cui, Y. Xu, and L. Liu, "SMAKA: Secure many-to-many authentication and key agreement scheme for vehicular networks," *IEEE Trans. Inf. Forensics Secur.*, vol. 16, pp. 1810–1824, 2021.

Guanjie Cheng received the BS degree from the School of Computer Science and Technology, Xidian University in 2018 and the PhD degree from the College of Computer Science and Technology, Zhejiang University in 2023. He is currently a researcher with the School of Software Technology in Zhejiang University, China. He was a visiting PhD in Nanyang Technological University, Singapore. His research interests lie in the fields of Internet-of-Things, data security, blockchain, and trusted management.

Yewei Wang received the bachelor's degree from the Department of Micro-nano Electronics, Shanghai Jiao Tong University, Shanghai, China. He was a cryptographic development engineer with Wanxiang Blockchain Labs. His research interests lie in secure multi-party computing, privacy computing, blockchain scalability, access control mechanisms, and trusted management.

Shuiguang Deng (Senior Member, IEEE) received the BS and PhD degrees in computer science from the College of Computer Science and Technology, Zhejiang University, China in 2002 and 2007, respectively. He is currently a full professor with the College of Computer Science and Technology in Zhejiang University, China. His research interests include edge computing, service computing, cloud computing, blockchain, and business process management. He serves for the journal *IEEE Transactions on Services Computing, Knowledge and Information Systems, Computing*, and *IET Cyber-Physical Systems: Theory & Applications* as an associate editor. Up to now, he has published more than 100 papers in journals and refereed conferences. He is a fellow of IET.

Zhengzhe Xiang (Member, IEEE) received the BS and PhD degrees in computer science and technology from Zhejiang University, Hangzhou, China and He was a visiting scholar in Shanghai Jiao Tong University, Shanghai, China. He is currently a lecturer with Hangzhou City University, Hangzhou, China. His research interests lie in the fields of service computing, cloud computing, and edge computing. He serves for several international journal like *IEEE Transactions on Mobile Computing*, *IEEE Transactions on Services Computing*, *Digital Communications* and *Networks* as reviewers.

Xueqiang Yan is currently a technology expert with the Wireless Technology Lab, Huawei Technologies. He was a member of technical staff with Bell Labs from 2000 to 2004. From 2004 to 2016, he was the director of the Strategy Department, Alcatel-Lucent Shanghai Bell. His current research interests include wireless networking, the Internet of Things, edge AI, future mobile network architecture, network convergence, and evolution.

Peng Zhao received the PhD degree in oncology from Sun Yat-sen university, Guangzhou, China, in 2007. He works with the Department of Medical Oncology, the First Affiliated Hospital, Zhejiang University School of Medicine, Hangzhou, China. He is currently leading some research projects supported by the National Natural Science Foundation of China. His research interests include artificial intelligence and edge computing in medical oncology.

Schahram Dustdar (Fellow, IEEE) is currently a full professor of computer science (Informatics) with a focus on Internet Technologies heading the Decentralized Systems Group with the TU Wien, Austria. He is recipient of the ACM Distinguished Scientist award (2009) and the IBM Faculty Award (2012). He is an associate editor of *IEEE Transactions on Services Computing*, *ACM Transactions on the Web*, and *ACM Transactions on Internet Technology* and on the editorial board of IEEE Internet Computing. He is the editor- in-chief of *Computing* (an SCI-ranked journal of Springer).