Blockchain-assisted Transparent Cross-domain Authorization and Authentication for Smart City

Cheng Huang, *Member, IEEE,* Liang Xue, *Student Member, IEEE,* Dongxiao Liu, *Member, IEEE,*
Xuemin (Sherman) Shen, *Fellow, IEEE,* Weihua Zhuang, *Fellow, IEEE,* Rob Sun, and Bidi Ying

***Abstract*—Secure cross-domain authorization and authentication (AA) enable application service providers (ASPs) to allow users for resource access from different trusted domains. In this paper, we propose a unified blockchain-assisted secure cross-domain AA framework for smart city, which can guarantee transparent cross-domain resource access while preserving user privacy. In the framework, ASPs can flexibly delegate their authentication capabilities to the blockchain, and users authorized by different ASPs can be authenticated by the blockchain where the authentication events are publicly audited and traced. Since the blockchain is publicly accessible, users' sensitive identity attributes may be exposed during the authentication process. To address privacy leakage caused by the authentication events, several privacy-preserving techniques, including threshold-based homomorphic encryption, zero-knowledge proof, and random permutation, are exploited to hide users' sensitive information on the blockchain. Moreover, to improve user revocation efficiency, we integrate a cryptographic accumulator and secure hash functions into the framework where ASPs are allowed to revoke their users through a global revocation contract. Our security analysis shows that the proposed framework can achieve all desirable security and privacy properties, and a proof-of-concept prototype has been developed to demonstrate the correctness and efficiency of the proposed framework.**

***Index Terms*—Smart city applications, cross-domain authorization and authentication, identity attribute privacy, decentralized trust, blockchain.**

I. Introduction

With the flourishing and advancement of ubiquitous sensing, wireless networking, and artificial intelligence technologies, the concept of smart city applications has been prevalent [1], where a smart city serves as a service organization while the citizens are considered to be the users of multiple smart city applications. The smart city applications include but are not limited to smart ehealthcare [2], location-aware services [3], vehicle-to-everything applications [4], smart grid [5], and mobile payment [6], which can provide a convenient, intelligent, and comfortable life for local residents. With the establishment of more smart city applications recently, the collaboration among multiple application service providers (ASPs) has become an inevitable trend for resources sharing and data exchanges [7], [8], [9]. For example, a vehicle-to-everything service such as navigation can collaborate with road monitoring services to obtain real-time road information to arrange a fast route for its users to reduce the travelling delay. Although the collaboration may bring advantages and benefits, it also leads to severe security and privacy concerns [10].

Among all security and privacy issues, cross-domain authorization and authentication (AA) are critical and have drawn much attention. In general, a smart city application has its own trusted domain and prevents all access from outside the domain without specific authorization and authentication. Only a user who has registered at the ASP can access its resources or functionalities with permission. When multiple applications work together to provide a joint service for users, the users may expose sensitive identity information and other private information to different ASPs during the authentication process, which potentially violating users' privacy requirements. For instance, a user may have to reveal their detailed driver license information issued by a transportation department to a car renting company for renting a car, which can unnecessarily expose their home addresses [11]. From another point of view, cross-domain AA can require a significant communication overhead among the ASPs when they run distributed protocols to authenticate unregistered external users [12]. Considering that multiple ASPs belonging to different trusted domains need to authenticate an unregistered external user, they need to communicate with each other in advance and make a distributed agreement on the user's identity attributes and permissions, which requires large communication costs.

Many solutions rely on a blockchain platform to achieve cross-domain AA in recent years, where ASPs from different trusted domains can exchange authorization information, and users can anonymously authenticate themselves to the ASPs afterwards [13], [14], [15]. The blockchain is usually regarded as an immutable and distributed ledger, and the ASPs belonging to different trusted domains can publish and store their public identity information on the blockchain. When receiving an external user's authentication request, an ASP can query the necessary information from the blockchain, such as the public keys associated with the user's identity credential, and verify the identity's validity. Under the circumstance, the cross-domain authentication interactions between the user and the ASPs are off-chain, which simplifies the design of a privacy-preserving cross-domain authentication solution. On one hand, these schemes need only to deal with privacy leakage between users and ASPs [16], without considering the effects caused by the blockchain. Therefore, these solutions can employ attribute-based anonymous credential techniques to achieve anonymous authentication in a straightforward way [17]. On the other hand, since the cross-domain authentication

This work was financially supported by Huawei Technologies Canada and the Natural Sciences and Engineering Research Council (NSERC) of Canada.

C. Huang, L. Xue, D. Liu, X. Shen, and W. Zhuang are with the Department of Electrical and Computer Engineering, University of Waterloo, Canada.

R. Sun and B. Ying are with the Huawei Technologies Canada, Ottawa, Ontario, Canada.

interactions are not recorded on the blockchain, their solutions lack transparency and accountability. Malicious and abnormal cross-domain authentication events can be ignored, which may lead to serious security vulnerabilities. As a result, the question arises: *Can we design a cross-domain AA framework based on the blockchain that simultaneously achieves transparency and privacy preservation?*

There exist many challenges to design such a unified framework. First, traditional attribute-based anonymous credential techniques support only selective disclosure in terms of privacy protection, and cannot be straightforwardly applied in the framework. When cross-domain authentication is executed on-chain, user identity attributes are exposed according to the authentication policy, which is not desirable. Neither the identity attributes of users nor the authentication policies of ASPs should be exposed during the on-chain authentication process. Second, the framework should be scalable and allow different ASPs and users to freely join and leave the system. At the same time, each ASP should be able to generate arbitrary authentication policies without interacting with others to improve communication efficiency. These functionalities have not been achieved in one unified framework so far, to the best of our knowledge. Third, conventional revocation schemes enable each ASP to maintain its own revocation list, which is not suitable for cross-domain AA scenarios. A global revocation mechanism should be in place for various applications to reduce the storage and query costs.

To meet the challenges, we introduce decentralized trust in our proposed framework. The cross-domain authentication interactions among different trusted domains are achieved with the assistance of decentralized authorities, i.e., a committee including a bunch of identity committee members. Each ASP can privately delegate its authentication capability to the committee using smart contracts, and the committee member can help verify whether an external user's identity credential matches the authentication policy set up by the ASP. The automated smart contract can regularize the behavior of committee members. When a member behaves maliciously, it can be identified and removed from the committee. To fulfill the security and privacy requirements, several cryptographic building blocks are applied in the framework. Specifically, we design a threshold-based partially homomorphic encryption scheme with the zero-knowledge proof technique to hide the user's identity attributes and the authentication policy during the on-chain authentication process. The hidden attributes are linked to authorized identity credentials based on the randomizable signature [18]. To improve the on-chain computational efficiency, we propose a new algorithm that encodes arbitrary authentication policies using dummy attributes. The algorithm can significantly reduce the costs of matching authentication policies, compared with the encoding method used in functional encryption [19]. Moreover, our proposed framework integrates a cryptographic accumulator (RSA accumulator) and secure hash functions to realize the revocation function through a global revocation contract. The revocation contract ensures that an ASP can revoke only the identity credential issued by itself but not identity credential issued by other ASPs. Figure 1 illustrates the main difference between our framework and the existing blockchain-based AA frameworks in terms of transparency and privacy preservation.

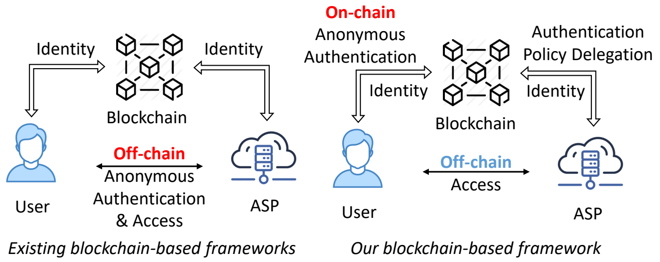


Figure 1: Comparison between existing blockchain-based AA frameworks and our proposed framework

The main contributions of this paper are summarized as follows:

- We propose a new unified secure cross-domain authorization and authentication framework for smart city applications based on the consortium blockchain. With the assistance of a decentralized identity committee, the proposed framework can achieve the transparency property for cross-domain authentication by deploying authentication contracts. The proposed framework is especially suitable for resource-constraint devices. Since all heavy authentication operations are outsourced to the decentralized identity committee, the off-chain authentication is lightweight and the cost is constant with respect to the complexity of authentication policies;
- The proposed framework is scalable and flexible, and supports many necessary functions, including convenient identity management and revocation. These functions are not fully achieved in existing authentication schemes, to the best of our knowledge. Furthermore, the proposed framework meets the security and privacy requirements for the cross-domain framework, and an ideal vs. real world simulation-based approach is utilized to prove its security properties;
- We have developed a proof-of-concept prototype of the proposed framework using Java language based on a consortium blockchain, Hyperledger Fabric. All the proposed protocols and algorithms are fully implemented based on the MIRACL library. The experiment results demonstrate the efficiency of the proposed on-chain and off-chain solutions.

The remainder of this paper is organized as follows. In Section II, some related works are reviewed and compared with our work. In Section III, we introduce the system model, define the adversary model, and identify the design goals. Then, we propose a secure and transparent cross-domain authorization and authentication framework based on blockchain in Section IV. Subsequently, security analysis and performance evaluation are presented in Sections V and VI, respectively. Finally, Section VII draws the conclusion of this work.

II. Related Works

In this section, we discuss recent studies which aim to achieve secure cross-domain authorization and authentication for smart city applications with privacy preservation.

Many AA solutions are proposed for single-domain smart city applications to satisfy security and privacy requirements [20], [21], [22]. Two common cryptographic primitives are usually adopted: the attribute-based signature (ABS) scheme and the attribute-based anonymous credential (ABAC) scheme [23], [24]. These two cryptographic tools and their variants are used not only for designing enterprise-level products but also in privacy-preserving AA solutions [25], [26]. The ABS scheme relies on one trusted identity issuer or multiple identity issuers to generate a private key associated with a set of identity attributes. A user can generate a unique signature using the private key and attest to the fact that its attributes satisfy some authentication policies [27], [28]. The ABAC scheme allows a user to prove that it possesses some identity attributes signed by one or multiple trusted identity issuers anonymously through the zero-knowledge proof of knowledge. Both techniques have their advantages and limitations, but they cannot be straightforwardly employed to achieve cross-domain AA.

To achieve secure cross-domain AA with privacy preservation, a recent new approach is to manage application domains through a blockchain platform. The decentralized platform can build trust among application domains via a distributed ledger. Combining with the cryptographic building blocks as mentioned, there exist blockchain-based secure cross-domain AA schemes with different focuses. Liu et al. propose a decentralized anonymous authentication scheme for space-ground networking applications [29]. Identity issuers are organized as a group on a blockchain platform during the authorization process, and a threshold-based randomizable signature scheme is presented. User identity attributes are signed by a group, and can be recognized by any group member to achieve cross-domain authentication. A similar cross-domain authentication architecture is proposed where authentication servers of various application domains can jointly authorize a user identity to achieve cross-domain authentication [30]. However, the above-mentioned two schemes cannot be applied to smart city scenarios due to limited flexibility, namely, identity issuers are not able to freely join or leave the system. Cheng et al. propose a blockchain-based mutual authentication scheme for collaborative edge computing services [14]. They use certificateless cryptography and elliptic curve cryptography to design a cross-domain authentication scheme where the blockchain is maintained as a public distributed ledger to store the global context among identity issuers. Although the scheme is efficient in terms of computational performance, it requires deploying an identity registration server that should be semi-trusted to guarantee security requirements. To avoid any trust in certificate-based solutions, a certificate transparency system is introduced into the blockchain platform to achieve cross-domain authentication, where any identity registration and revocation operations can be traced on the blockchain to eliminate a centralized trusted certificate management server [31]. Different from certificate-based solutions, identity-based signature (IBS) is another approach that can be integrated with the blockchain platform. Based on the IBS, a secure cross-domain AA scheme can manage and authenticate user identity on the blockchain platform where public certificates are represented using entities' identities [13].

TABLE I: Comparison between existing cross-domain AA schemes and our work

Reference	[29]	[30]	[14]	[31]	[13]	[32]	[15]	Our Work
Cross-domain AA	✓	✓	✓	✓	✓	✓	✓	✓
Identity Privacy	✓	✖	✓	✓	✓	✖	✖	✓
Multi-attribute Support	✓	✖	✖	✖	✖	✖	✓	✓
Flexibility	✖	✖	✓	✓	✓	✓	✓	✓
Non-interactivity	✓	✓	✓	✖	✓	✓	✖	✓
Auth Transparency	✖	✖	✖	✖	✖	✓	✓	✓
Revocation	✖	✖	✓	✖	✖	✓	✓	✓

Beyond a distributed ledger, the blockchain platform can be regarded as a distributed Turing machine to achieve secure cross-domain authentication and offer the authentication transparency property. Zhang et al. propose a secure cross-domain authentication based on a blockchain platform where an authentication contract is deployed to automatically verify any user identity for independent application domains and reduce the computational cost at the authentication server [15]. Nevertheless, the privacy of the user identity is not protected, i.e., any user authentication can be linked publicly, which may not be suitable for application scenarios under restricted privacy regulations. Ali et al. propose a decentralized blockchain-assisted permission delegation framework [32]. In this framework, identity issuers can issue permissions to users through a local and global delegation policy smart contract, and identity verifiers can read the on-chain delegation policy to determine whether a user has permission to access their resources. Similar to the scheme in [15], all authentication behaviors are published on the blockchain, which may violate the privacy protection demands.

To highlight the differences between the existing schemes and our newly proposed solutions, we compare them in Table. I in terms of seven objectives, including cross-domain AA, identity privacy, multi-attribute support, flexibility, non-interactivity, authentication transparency, and revocation (Detailed description of these objectives is given in Section III). These objectives are essential in achieving practical secure cross-domain authentication for smart city applications and our work achieves all of them. Notice that existing identity privacy has different definitions: pseudonymity is sufficient in some definitions, while selective disclosure of attributes should be achieved in others. However, in our proposed framework, neither pseudonymity nor selective disclosure is sufficient. A more rigorous privacy definition is necessary, since we consider a unified framework where all authentication interactions are transparent and can be audited. Not only should the user attributes be protected, but also the authentication policy should be hidden from a user if the user identity credential does not match the policy.

III. Models and Design Goals

In this section, we define the system model of identity management, authorization, and authentication for smart city applications, and identify the adversarial model and the design goals.

A. System Model

As shown in Figure 2, the system under consideration mainly consists of five types of entities, namely decentralized identity committee members, identity issuers, identity holders, identity verifiers, and an identity auditor:

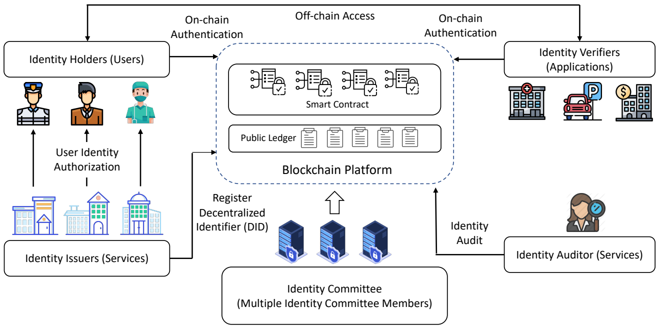

Off-chain Access
On-chain Authentication
On-chain Authentication
Identity Holders (Users)
Identity Verifiers (Applications)
Smart Contract
Public Ledger
Blockchain Platform
User Identity Authorization
Register Decentralized Identifier (DID)
Identity Audit
Identity Issuers (Services)
Identity Auditor (Services)
Identity Committee (Multiple Identity Committee Members)

Figure 2: The system model

- Identity committee members ($ICMs$) are organized to be one identity committee that is responsible for constructing a consortium blockchain platform where other entities can send transactions and publish smart contracts as distributed computer programs. The major functionality of this permissioned blockchain is to assist the cross-domain identity management and the transparent authentication for diverse smart city applications. Such applications need to work together to be functional, e.g., achieving non-interactive cross-domain authentication, storing the context of identity authentication information, and maintaining detailed logs for future identity auditing. In reality, they are cloud servers controlled by independent authorities;
- Identity issuers (IIs) are publicly-recognized companies or organizations that can authorize identity holders by issuing identity credentials. An identity credential can be viewed as a collection of identity attributes that distinguish an identity holder from other entities. For various applications including ehealthcare and V2X services, there may exist multiple identity issuers that belong to a large number of application domains;
- Identity holders (IHs) can be viewed as users[1] who apply for identity credentials to meet authentication requirements. The holders' backgrounds need to be verified by identity issuers before receiving the corresponding identity credentials who need to store these identity credentials locally in their credential databases, e.g., their personal smartphones. By showing the identity credential in a privacy-preserving way, an identity holder can achieve on/off-chain identity authentication by interacting with the blockchain and identity verifiers;
- Identity verifiers (IVs) are application service providers whose goal is to verify the identity of an identity holder, in order to determine whether the identity satisfies a predefined authentication policy. Each authentication policy is written into a smart contract in a privacy-preserving manner and is uploaded to the blockchain platform. It is worth noting that an identity verifier can be the same entity as an identity issuer or a different entity;
- Identity auditor (IA) is controlled by a trusted regulatory department following the data protection regulation law. It does not participate in the authentication process but is deployed in the system to trace each authentication event and the real identity of the identity holder with the help of identity issuers and identity committee members.

B. Adversarial Model

We consider a static malicious adversarial model in the system. The majority of identity committee members are honest, while others can be compromised. The malicious external adversaries, identity issuers, identity holders, identity verifiers may also exist and collude with each other, who can arbitrarily behave to break the system in the following attacks:

- Privacy attack - The adversaries aim to break the identity privacy guarantees. They can perform arbitrarily to compromise the identity information, including identity attributes of an honest identity holder, and identify the identity holder during the authentication interactions;
- Forgery attack - The adversaries aim to forge their identity credentials and pretend to be a valid identity holder to pass the identity authentication. If a valid identity holder already exists and can be forged, the forgery attack is a special impersonation attack;
- Replay attack - The adversaries aim to replay an existing authentication interaction to pass the identity authentication;
- Repudiation attack - The adversaries aim to deny an existing authentication interaction with an identity verifier.

To capture these malicious attacks in a unified model, ideal/real world is defined, and a simulation-based proof [33] is given in the security analysis to prove that these attacks can be prevented, and the security properties are guaranteed. It is assumed that the identity auditor is a trusted entity since it should be able to trace the real identity of any identity holder if necessary. Under such an assumption, the security can be further extended by substituting one identity auditor with decentralized identity auditors [34]. Other attacks such as denial of service attacks are not considered here.

C. Design Goals

Based on the system model and adversarial model, we aim to achieve the following design goals:

- Cross-domain AA - Identity verifiers can easily verify any identity holder's attributes that authorized by different identity issuers;
- Flexible identity management - Identity issuers, holders, and verifiers are free to join and leave the system without communicating with other entities;
- Non-interactivity - For identity authentication, arbitrary authentication policies can be set by an identity verifier, i.e., the identity verifier can generate an authentication

[1] The terms "identity holder" and "user" are interchangeable for easy understanding in this paper.

policy that contains identity attributes managed by different identity issuers without interacting with them;

- Authentication transparency - All authentication events and interactions are transparent and can be traced if necessary;
- Easy revocation - In case that some identity credentials are not available, any issued identity credential should be allowed to be revoked efficiently by its issuer;
- Security and privacy - The system should follow the general privacy protection regulation and resist the malicious attacks defined in the adversarial model;
- Low verification cost - The verification cost is relatively low at the side of identity verifiers, and the computational cost should be acceptable in terms of on-chain operations.

IV. Proposed Transparent Cross-domain Authorization and Authentication Framework

In this section, we propose a new blockchain-assisted transparent cross-domain authorization and authentication framework, which consists of seven stages: 1) system setup, 2) service entity registration, 3) user registration and credential authorization, 4) authentication policy generation and contract deployment, 5) on/off-chain authentication, 6) identity auditing, and 7) credential revocation.

A. Main Ideas

We give an overview of the proposed framework, discuss the main ideas, and highlight the novelty of the proposed framework before diving into details. In existing blockchain-based cross-domain identity management [35], authorization, and authentication frameworks, blockchain is utilized as a public distributed ledger to store the identity issuers' decentralized identifiers (a.k.a public keys), while authentication interactions between identity holders and identity verifiers are totally off-chain. In contrast, the proposed framework allows an identity verifier to delegate most of the authentication procedures to the blockchain through a well-designed authentication smart contract that encodes any kind of authentication policy. By doing so, any authentication request triggered by an identity holder is transparent and can be authenticated on-chain without the participation of identity verifiers. The security of the authentication contract is determined by the decentralized trust of identity committee members. Since most of the authentication procedures are outsourced to the blockchain, the computational efficiency can be significantly improved for identity verifiers. In the meantime, the authentication contract supports identity auditing, and an additional revocation contract is deployed to achieve flexible and easy credential revocation for invalid identity credentials. However, the transparency is a double-edged sword due to privacy concerns of authentication information, which is an inevitable challenge.

To achieve both transparency and privacy preservation, the proposed framework relies on new authorization and authentication protocols, which are an extension and hybrid of cryptographic primitives including randomizable signature, threshold-based partial homomorphic encryption, zero-knowledge proof of knowledge, and RSA accumulator. The randomizable signature can be used for constructing a privacy-preserving identity registration and credential insurance protocol in our proposed framework. Due to the unforgeability property of the randomizable signature, users' identity credentials are also unforgeable, which fulfills our security requirements. The threshold-based partial homomorphic encryption is for hiding the identity attributes during the on-chain and off-chain authentication interactions in a distributed setting. Since the threshold cryptosystem also has the homomorphic property, users' attributes and services' authentication policies can be encrypted and matched in ciphertext to guarantee our privacy requirement during the authentication. The zero-knowledge proof of knowledge can help link the randomizable signature to the encrypted identity attributes. With the zero-knowledge proof, users cannot cheat in the authentication process to claim some identity attributes that do not belong to them, even though the attributes are hidden. By integrating the RSA accumulator with the zero-knowledge proof, the proposed framework can additionally achieve flexible identity revocation. Revoked identity credentials can be accumulated into a revocation list by the identity issuers, and users can use the zero-knowledge proof to prove that they are not revoked by proving their credentials do not exist in the revocation list. Moreover, to further improve the efficiency of on/off-chain authentication, we propose a dummy-attribute-based authentication policy encoding method for on/off-chain authentication. Compared with the conventional policy encoding method [19], the computational cost of the proposed method remains almost constant regardless of the number of encoded credential attributes.

B. Detailed Construction

System Setup: In this stage, security parameters are generated by identity committee members, and a consortium blockchain is established.

Specifically, given a security parameter, τ, bilinear group parameters $pp = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, g_1, g_2)$ are generated, where p is a large safe prime whose length is τ and is the order of three multiplicative cyclic groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$, $e : G_1 \times G_2 \to G_T$ is a type-III asymmetric bilinear map, g and g_1 are two distinct generators of G_1, and g_2 is a generator of G_2.

Supposing that there exist $\mathcal{N}$ identity committee members $\{ICM_1, ICM_2, ..., ICM_{\mathcal{N}}\}$ who run a secure distributed key generation protocol in a synchronous setting to generate a shared public key $spk = g_1^s$ [36]. Each committee member's private key $sk_i \in Z_p^*$ satisfies a polynomial $sk_i = f(i) = s + \Sigma_{j=1}^{T-1} \mathsf{coff}_j \cdot (i)^j \bmod p$. Here, coff_j is randomly chosen by decentralized committee members, and T indicates the security level of decentralized trust. Each committee member, ICM_i, also publishes and proves the auxiliary information, $aux_i = g^{sk_i}$. Furthermore, these committee members run a distributed RSA key generation protocol to generate an RSA group $(\hat{N}, \hat{g}, \hat{h})$ [37]. In public RSA parameters $(\hat{N}, \hat{g}, \hat{h})$, $\hat{N} = \hat{p}\hat{q}$ is the RSA modulus and $\hat{p}$ and $\hat{q}$ are large safe primes; $\hat{N}$'s bit length $\hat{\tau}$ determines the security level of the RSA group; $\hat{g} \in QR_{\hat{N}}$ and $\hat{h} \in QR_{\hat{N}}$ are two distinct generators of $Z_{\hat{N}}^*$.

These committee members construct a consortium blockchain platform, e.g., Hyperledger Fabric [38], and they do not fully trust each other but a majority of them are believed to be honest. Let H, H_1, and H_2 be three cryptographic hash functions, each satisfying $H : \{0,1\}^* \rightarrow Primes(2^{l-1}, 2^l)$, $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1$, and $H_2 : \{0,1\}^* \rightarrow \mathbb{G}_2$, with l being the bit length of the prime number. Finally, bilinear group parameters pp, RSA public parameters $(\hat{N}, \hat{e}, \hat{g}, \hat{h})$, shared public key spk, hash functions (H, H_1, H_2), auxiliary information (i, ICM_i, aux_i) of each identity committee member, and two additional security parameters τ_1 and τ_2 are written into the genesis block of the blockchain platform. Security parameters τ_1 and τ_2 are the statistical zero-knowledge and soundness security parameters respectively of the proposed zero-knowledge proof protocols. Note that $\tau - 1 > l$ and $\tau - 1 > \tau_1 + \tau_2$ should be satisfied for the security requirements.

Service Entity Registration: In this stage, identity issuers, identity verifiers, and the identity auditor can register themselves on the blockchain as valid service entities. For example, identity issuers can be government departments such as traffic administration bureau, and verifiers can be application companies. In some cases, the identity issuers can also be the identity verifiers, e.g., a bank service provider can be either an identity issuer or a verifier. We omit the details of on-chain entity registration since the registration is simple and straightforward. Each entity only needs to publish a public key as its public address, and keeps a private key for generating its signature on any transaction data.

Identity issuers and the identity auditor also need to publish their decentralized identifiers to the blockchain since they have to be identified by users according to their services. Each identity issuer, $II_i \in \{II_1, II_2, ...\}$, reads the public parameters in the genesis block of the blockchain, and chooses $K_i + 3$ random numbers $(x_i, y_{i0}, y_{i1}, ..., y_{iK_i}, z_i) \in Z_p^{K_i+3}$, where K_i is the maximum number of identity attribute categories that II_i wants to authorize, and may be different for identity issuers. Each attribute belongs to an attribute category, e.g., an address is an attribute category while people have various addresses. The issuer then generates public and private key pairs (pk_i', sk_i'), where $pk_i' = (Y_{i0} = g_1^{y_{i0}}, Y_{i1} = g_1^{y_{i1}}, ..., Y_{iK_i} = g_1^{y_{iK_i}}, Z_i = g_1^{z_i}, X_i' = g_2^{x_i}, Y_{i0}' = g_2^{y_{i0}}, Y_{i1}' = g_2^{y_{i1}}, ..., Y_{iK_i}' = g_2^{y_{iK_i}}, Z_i' = g_2^{z_i})$ and $sk_i' = X_i = g_1^{x_i}$. Public key pk_i' is published as a decentralized identifier (DID) to the blockchain's public ledger and non-interactive zero-knowledge proof of knowledge π_{II_i} is published correspondingly:

$$\begin{aligned}\pi_{II_i} \leftarrow & ZkPoK\{(x_i, y_{i0}, y_{i1}, ..., y_{iK_i}, z_i) : Y_{i0} = g_1^{y_{i0}}, \\ & Y_{i1} = g_1^{y_{i1}}, ..., Y_{iK_i} = g_1^{y_{iK_i}}, Z_i = g_1^{z_i}, X_i' = g_2^{x_i}, \\ & Y_{i0}' = g_2^{y_{i0}}, Y_{i1}' = g_2^{y_{i1}}, ..., Y_{iK_i}' = g_2^{y_{iK_i}}, Z_i' = g_2^{z_i}\}.\end{aligned}$$

The identity auditor generates public/private key pair $\tilde{pk} = g^z$ and $\tilde{sk} = z \in Z_p^*$ which is chosen randomly. The auditor publishes public key $\tilde{pk}$ to the blockchain's public ledger as a DID and non-interactive zero-knowledge proof of knowledge π_{IA} is also published correspondingly:

$$\pi_{IA} \leftarrow ZkPoK\{(z) : \tilde{pk} = g^z\}.$$

These entities are free to leave the system by broadcasting that they are not available in the blockchain platform.

User Registration and Credential Authorization: In this stage, an identity holder, i.e, a user, can register at the identity issuer, II_i, who can generate a corresponding identity credential for the identity holder. Before sending the registration request, the identity holder needs to pass an identity background check and can create only one valid identity credential. The proof of identity uniqueness can be achieved in a privacy-preserving way according to a recent work [39] or in a plaintext format, e.g., a unique social insurance number. If duplicated identities are found or the background check is not approved, the registration halts.

The identity holder then chooses a random number, $t_i \in Z_p^*$, picks a unique serial number $u_i \in Z_p^*$, and computes a commitment as $\mathsf{Com}_i = g_1^{t_i} Y_{i0}^{u_i} \prod_{j=1}^{K_i} Y_{ij}^{a_{ij}}$. Identity attributes of the holder, $attrs_i = (a_{i1}, a_{i2}, ..., a_{iK_i})$, can be set to 0 in case that the attribute does not belong to the holder. Next, the holder sends $(\mathsf{Com}_i, attrs_i, \pi_{\mathsf{Com}_i})$ to the issuer, where π_{Com_i} is a non-interactive zero-knowledge proof of knowledge as follows. This proof is to prove that valid attributes are correctly embedded into the authorization request:

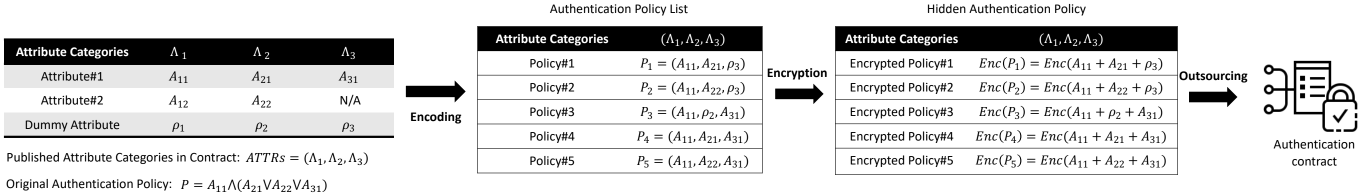
$$\pi_{\mathsf{Com}_i} \leftarrow ZkPoK\{(t_i, u_i) : \mathsf{Com}_i = g_1^{t_i} Y_{i0}^{u_i} \prod_{j=1}^{K_i} Y_{ij}^{a_{ij}}\}.$$

The issuer verifies attributes $attr_i$ and proof π_{Com_i}. If the proof is not valid or the attributes do not match the background of the issuer, the issuer returns with errors and the registration halts. Otherwise, the issuer generates a unique identifier $UID_i = H(pk_i'||ts||ind)$ for the holder where ts is the current timestamp and ind is the index of registered users, chooses a random number v_i, calculates $\sigma_i' = (\sigma_{i1}' = g_1^{v_i}, \sigma_{i2}' = (X_i \cdot \mathsf{Com}_i \cdot Z_i^{UID_i})^{v_i})$, and returns (σ_i', UID_i) to the holder. The identity issuer also stores $RID_i = g^{UID_i}$ into its local database. The holder unblinds σ_i' to obtain $\sigma_i = (\sigma_{i1} = g_1^{v_i}, \sigma_{i2} = \frac{\sigma_{i2}'}{\sigma_{i1}^{t_i}})$, and checks whether the following equation holds:

$$\begin{aligned}& e(\sigma_{i1}, X_i') \cdot e(\sigma_{i1}, Y_{i0}')^{u_i} \cdot \prod_{j=1}^{K_i} e(\sigma_{i1}, Y_{ij}')^{a_{ij}} \\ & \cdot e(\sigma_{i1}, Z_i')^{UID_i} = e(\sigma_{i2}, g_2).\end{aligned}$$

If the equation holds, the identity holder stores the identity credential authorized by II_i as $Cred_i = (u_i, UID_i, attrs_i, \sigma_i)$ locally in its credential database. This identity credential is a variant of a randomizable signature to ensure that the credential cannot be forged by malicious users.

Authentication Policy Generation and Contract Deployment: In this stage, an identity verifier generates authentication policies and deploys corresponding authentication smart contracts to the blockchain. An identity verifier may need to authenticate different identity attributes $(A_{11}, A_{21}, ...)$ that satisfy authentication policy $P = (A_{11} \wedge A_{21}) \vee A_{31} \wedge ...)$ where the relationship can be logic OR or AND. Here, $(A_{11}, A_{21}, ...)$ may belong to various identity attribute categories $\mathsf{ATTRs} = (\Lambda_1, \Lambda_2, ...)$, i.e., $A_{11} \in \Lambda_1, A_{21} \in \Lambda_2$.



Attribute Categories	Λ_1	Λ_2	Λ_3
Attribute#1	A_{11}	A_{21}	A_{31}
Attribute#2	A_{12}	A_{22}	N/A
Dummy Attribute	ρ_1	ρ_2	ρ_3

Published Attribute Categories in Contract: $ATTRs = (\Lambda_1, \Lambda_2, \Lambda_3)$

Original Authentication Policy: $P = A_{11} \wedge (A_{21} \vee A_{22} \vee A_{31})$

Authentication Policy List

Attribute Categories	$(\Lambda_1, \Lambda_2, \Lambda_3)$
Policy#1	$P_1 = (A_{11}, A_{21}, \rho_3)$
Policy#2	$P_2 = (A_{11}, A_{22}, \rho_3)$
Policy#3	$P_3 = (A_{11}, \rho_2, A_{31})$
Policy#4	$P_4 = (A_{11}, A_{21}, A_{31})$
Policy#5	$P_5 = (A_{11}, A_{22}, A_{31})$

Hidden Authentication Policy

Attribute Categories	$(\Lambda_1, \Lambda_2, \Lambda_3)$
Encrypted Policy#1	$Enc(P_1) = Enc(A_{11} + A_{21} + \rho_3)$
Encrypted Policy#2	$Enc(P_2) = Enc(A_{11} + A_{22} + \rho_3)$
Encrypted Policy#3	$Enc(P_3) = Enc(A_{11} + \rho_2 + A_{31})$
Encrypted Policy#4	$Enc(P_4) = Enc(A_{11} + A_{21} + A_{31})$
Encrypted Policy#5	$Enc(P_5) = Enc(A_{11} + A_{22} + A_{31})$

Figure 3: Outsourced authentication policy generation with dummy attributes

Algorithm 1 Check$(Shares[aid], Req[aid])$

```
Input: shared secret $Shares[aid]$ and authentication request $Req[aid]$;
Output: authentication result $Rst$;
$Rst = false$;
extract $\mathbb{V} = (V'_1, V'_2, ..., V'_{K'})$ from $Req[aid]$;
for $k = 1$ to $K'$ do
    $T_k = e(V'_{k1}, H_2(spk||ts))$;
    extract $T_{ik}$ from $Shares[aid]$ for $i = 1$ to $T$;
    $\zeta_i = \prod_{j=1, j\neq i}^{T} \frac{j}{j-i}$
    if $\prod_{i=1}^{T} (T_{ik})^{\zeta_i} == T_k$ then
        $Rst = true$;
        break;
    end if
end for
Output $Rst$;
```

Authentication Policy Generation: In the framework, an authentication policy, P, is encoded into an authentication policy list with dummy attributes $(P_1, P_2, ..., P_{K'})$, as shown in Figure 3. Each attribute category, Λ_j, is assigned with dummy attribute ρ_j, which can be authorized to everyone who requests the attribute. By doing so, even if an identity holder without a valid attribute that belongs to Λ_j can generate a valid authentication request which is indistinguishable from the holder with a valid attribute. In the meantime, the dummy attribute does not affect the correctness of an authentication interaction. To reduce storage cost of the authentication policy and ensure confidentiality of the authentication policy, the policies are aggregated and encrypted based on a variant of Elgamal encryption algorithm to create a hidden authentication policy. Concretely, the verifier first needs to compute an encrypted base, B, for the authentication policy. The encrypted base can be used by any identity holder to generate encrypted identity attributes according to the authentication policy (how an identity holder uses B to generate encrypted identity attributes is given in Algorithm 2, step-7). That is, the verifier picks two random numbers $\beta \in Z_p^*$ and $\gamma \in Z_p^*$, and calculates $B = (B_0, B_1) = (g_1^\beta, spk^\beta \cdot g_1^\gamma)$.

Then, to encrypt a policy, $P_k = (A_{11}, A_{21}, ...)$ for $k = 1, 2, ..., K'$, in the authentication policy list, the identity verifier aggregates the policy as $\Xi_k = \sum_{A \in P_k} A$, chooses random numbers $\eta_k \in Z_p^*$ and encrypts authentication policy P_k as

$$\begin{aligned} C_k &= Enc(P_k) = Enc(\Xi_k) = (C_{k0}, C_{k1}) \\ &= (B_0^{\Xi_k} \cdot g_1^{\eta_k}, B_1^{\Xi_k} \cdot spk^{\eta_k}) = (g_1^{\alpha_k}, spk^{\alpha_k} \cdot g_1^{\gamma \Xi_k}). \end{aligned}$$

The verifier proves that the ciphertext is correctly formed by providing the following non-interactive zero-knowledge proof of knowledge π_B. The proof is to prevent a malicious verifier from extracting the identity attributes of a user if their identity attributes do not match the authentication policy:

$$\begin{aligned} \pi_B \leftarrow & ZkPoK\{(\beta, \gamma, \eta_1, \eta_2, ..., \eta_{K'}, \Xi_1, \Xi_2, ..., \Xi_{K'}) : \\ & B_0 = g_1^\beta \wedge B_1 = spk^\beta \cdot g_1^\gamma \wedge \\ & \{C_{k0} = B_0^{\Xi_k} \cdot g_1^{\eta_k}\}_{k=1}^{K'} \wedge \{C_{k1} = B_1^{\Xi_k} \cdot spk^{\eta_k}\}_{k=1}^{K'}\}. \end{aligned}$$

Encrypted base B, its proof π_B, and encrypted authentication policies C_k for $k = 1, 2, ..., K'$, are stored as $\mathbb{C} = (C_1, C_2, ..., C_{K'})$.

Authentication Contract Deployment: The pseudocode of the authentication contract is given in Contract 1, which involves six functions: *Init*, *Auth*, *Vote*, *Query*, *Record*, and *Audit*. Each function, except function *Init*, can be triggered after accepting a transaction with/without parameters. By triggering function *Init* when deploying the contract, the verifier can set T identity committee members for cross-domain authentication, set the attribute categories ATTRs, and initialize some variables and parameters $params$, including bilinear parameters pp, public key spk, and auditing public key $\tilde{pk}$, and hidden authentication policy $\mathbb{C}$, encrypted base B, and proof of the authentication policy π_B. Function *Auth* can be triggered by a transaction sent from an identity holder. The transaction consists of authentication identifier aid, randomized identity credential σ, encrypted authentication request auth, traceable identifier ID, non-interactive zero-knowledge proof of knowledge π_{auth}, and off-chain short-term authentication token Δ. Function *Vote* is triggered by transactions sent by identity committee members to vote for a final authentication result. The transaction consists of aid and the index of identity committee member i, secret share ss_i, non-interactive zero-knowledge proof of knowledge π_{ss_i}, and tag information tag indicating whether the authentication request is acceptable or not. In this function, ZkVerify$_{\text{rec}}$ is a verification algorithm that verifies the validity of π_{ss_i} and Check is an authentication result finalizing algorithm as shown in Algorithm 1. Function *Query* can be triggered publicly for querying the authentication result, and function *Record* can be triggered by the identity verifier for recording local off-chain authentication and access attempts for the purpose of auditing. When recording the attempts, timestamp ts and information info related to the description of accessed resources and permissions should be stored. Function *Audit* can be triggered by the identity auditor to trace the real identity of the user by sending a transaction

```
Init:   Set $ICMs := \{ICM_1, ICM_2, ..., ICM_T\}$;
        Set $\mathsf{ATTRs} := (\Lambda_1, \Lambda_2, ..., )$
        Set $params := \{pp, spk, \tilde{pk}, \mathbb{C}, B, \pi_B\}$;
        Set $Rec = \{\}$, $Num := \{\}$, and $State := \{\}$;
        Set $TmpCreds := \{\}$ and $Votes := \{\}$;
        Set $Req := \{\}$ and $Rsp := \{\}$;
        Set $\mathsf{IDs} = \{\}$, $Shares := \{\}$, and $Rst := \{\}$;
Auth:   Upon receiving from $IH(aid, \sigma, \mathsf{auth}, ID, \pi_{auth}, \Delta)$
          Assert $0 = State[aid]$;
          $Req[aid] := \mathsf{auth}$;
          $\mathsf{IDs}[aid] := ID$;
          $TmpCreds[aid] := \Delta$;
          $Num[aid] := 0$;
          $Rst[aid] := false$;
          $State[aid] := 1$;
          $Votes[aid] = 1$;
Vote:   Upon receiving from $ICM_i(aid, i, ss_i, \pi_{ss_i}, tag)$
          Assert $1 = State[aid]$;
          If $T = Num[aid]$ and $1 = Votes[aid]$ then
            $Rst[aid] := \mathsf{Check}(Shares[aid], Req[aid])$;
            $State[aid] := 2$;
          Else
            Assert $0 = ICM_i[aid].Submit()$;
            If $tag == 1$ then
              Assert $1 = \mathsf{ZkVerify}_{\mathsf{rec}}(aux_i, ss_i, \pi_{ss_i})$;
              $Num[aid] := Num[aid] + 1$;
              $Shares[aid][i] := ss_i$;
            Else
              $Votes[aid] = 0$;
Query:  Upon receiving from $IH, ICM_i, IV(aid)$
          Assert $State[aid] = 2$;
          return $(Rst[aid], TmpCreds[aid])$;
Record: Upon receiving from $IV(aid, ts, \mathsf{info})$
          Assert $2 = State[aid]$;
          $Rec[aid].push(aid, ts, \mathsf{info})$;
Audit:  Upon receiving from $IA(aid)$
          Assert $2 = State[aid]$;
          return $\mathsf{IDs}[aid]$;
```

Contract 1: Pseudocode of the authentication contract

including authentication identifier aid. If an authentication exists, encrypted identifier, $\mathsf{IDs}[aid]$, will be returned.

After the contract is successfully uploaded to the blockchain platform, the identity committee members need to monitor the authentication contract such that they can respond to the on-chain authentication in the next phase. Namely, they should be able to track the changing states of an on-chain authentication request, and participate in the on-chain authentication interactions through a monitoring service deployed in their servers.

On/off-chain Authentication: The on/off-chain identity authentication stage includes three sub-phases: In sub-phase-1, an identity holder anonymously authenticates itself through a web/mobile application to access the permissioned blockchain; in sub-phase-2, the holder locates the authentication contract at the blockchain and achieves on-chain authentication through communicating with the blockchain; and in sub-phase-3, the holder achieves off-chain authentication with the identity verifier. Sub-phase-1 and sub-phase-2 are pre-authentication phases which do not need to be real-time, and sub-phase-3 is a real-time off-chain authentication interaction between the user and the identity verifier in order to access some particular resources. The identity holder can execute sub-phase-1 and sub-phase-2 in advance and later run sub-phase-3. Considering that identity holders want to authenticate themselves to an identity verifier, they just execute sub-phase-3 in real time to significantly reduce the authentication delay caused by complex zero-knowledge proofs on identity attributes. Figure 4 shows the three sub-phases and main procedure of the on/off-chain authentication.

In sub-phase-1, identity holders achieve anonymous authentication by proving that they possess one valid identity credential issued by a registered identity issuer. After the authentication is passed, the holder can access the blockchain through a web client or mobile client published by the committee. Specifically, for identity credential $Cred_i$, the holder chooses two random numbers $v_i \in Z_p^*$ and $t_i \in Z_p^*$, and randomizes σ_i as:

$$\sigma_i' = (\sigma_{i1}', \sigma_{i2}') = (\sigma_{i1}^{v_i}, (\sigma_{i2} \cdot \sigma_{i1}^{t_i})^{v_i}).$$

The holder then generates non-interactive zero-knowledge proof of knowledge π_σ, and sends (σ_i', π_σ) to an identity committee member:

$$\begin{aligned} \pi_\sigma \leftarrow & ZkPoK\{(u_i, a_{i1}, a_{i2}, ..., a_{iK_i}, UID_i, t_i) : \\ & e(\sigma_{i1}', X_i') \cdot e(\sigma_{i1}', Y_{i0}')^{u_i} \cdot \prod_{j=1}^{K_i} e(\sigma_{i1}', Y_{ij}')^{a_{ij}} \\ & \cdot e(\sigma_{i1}', Z_i')^{UID_i} \cdot e(\sigma_{i1}', g_2)^{t_i} = e(\sigma_{i2}', g_2)\}. \end{aligned}$$

The identity committee member verifies proof π_σ. If the proof is valid, it means that the client user is a valid identity holder; thus the committee member allows the holder to access the blockchain. Otherwise, it rejects the access request. For example, the web/mobile client published by the identity committee can be regarded as an interface between an identity holder and the blockchain. The purpose of the interface is to hide the holder's on-chain identity, as all transactions sent from the user is actually signed by the identity committee member.

In sub-phase-2, after locating the authentication contract published by a particular identity verifier (i.e., an application), the identity holder can achieve on-chain authentication. For illustration purposes, a simple case is considered here. The identity holder possesses only one identity credential, $Cred_i$, from identity issuer II_i.

The on-chain authentication protocol does not need participation of the identity verifier, and the identity holder needs to interact with only the blockchain platform. The identity holder executes Algorithm 2 to generate $(aid, \sigma, \mathsf{auth}, ID, \pi_{auth}, \Delta)$ and sends transaction $(aid, \sigma, \mathsf{auth}, ID, \pi_{auth}, \Delta)$ to trigger function *Auth* in the authentication contract. In step-2 of the algorithm, σ_i is randomized such that the holder's identity credential, $Cred_i$, cannot be linked to preserve anonymity property; in step-3, the holder downloads revocation information RL and Q that are utilized in step-18 and step-21 to prove it is not revoked; $attr_i'$, W_i, $\mathbb{C}'$, and $\mathbb{V}$ in step-7, step-9, step-11, and step-12 are intermediate results for generating ciphertext of final matching results $\mathbb{V}'$; $attr_i' = (U_{i1}, U_{i2}, ..., U_{iK_i})$ is the ciphertext of the holder's identity attributes, where

$$\begin{aligned} U_{ij} &= (U_{ij0}, U_{ij1}) = (B_0^{\epsilon a_{ij}} \cdot g_1^{r_j}, B_1^{\epsilon a_{ij}} \cdot spk^{r_j}) \\ &= (g_1^{\beta \epsilon a_{ij} + r_j}, g_1^{\epsilon \gamma a_{ij}} spk^{\epsilon \beta a_{ij} + r_j}); \end{aligned}$$

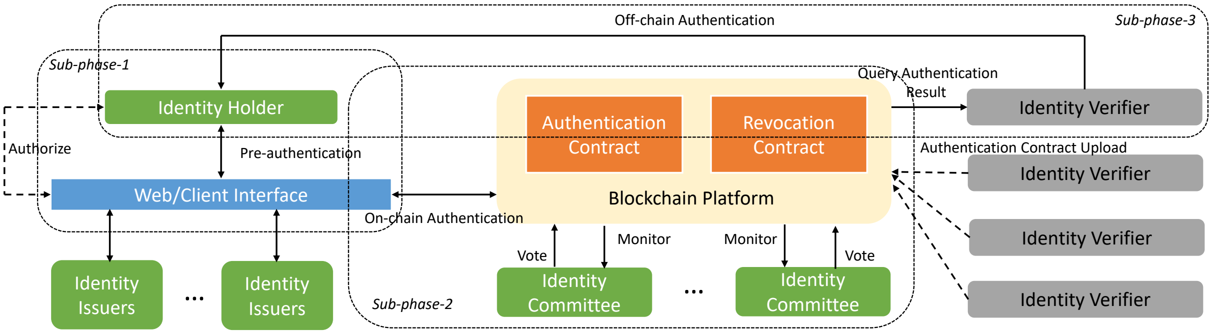


Figure 4: The three sub-phases for on/off-chain authentication

$$\begin{aligned}\pi_{auth} \leftarrow & ZkPoK\{(u_i, a_{i1}, ..., a_{iK_i}, \epsilon, r_1, ..., r_{K_i}, \theta_1, \theta_2, ..., \theta_{K'}, UID_i, r, \hat{r}, t_i, \tilde{r}, D, \omega) : \\ & ID_1 = g^{\tilde{r}} \wedge ID_2 = g^{UID_i} \tilde{pk}^{\tilde{r}} \wedge \{V_k = (V_{k0}, V_{k1}) = (W_{i0} \cdot C'_{k0}, W_{i1} \cdot C'_{k1})\}_{k=1}^{K'} \wedge \\ & e(\sigma'_{i1}, X'_i) \cdot e(\sigma'_{i1}, Y'_{i0})^{u_i} \cdot \prod_{j=1}^{K_i} e(\sigma'_{i1}, Y'_{ij})^{a_{ij}} \cdot e(\sigma'_{i1}, Z'_i)^{UID_i} \cdot e(\sigma'_{i1}, g)^{t_i} = e(\sigma'_{i2}, g_2) \wedge \epsilon \neq 0 \wedge \\ & \{U_{ij} = (U_{ij0}, U_{ij1}) = (B_0^{\epsilon a_{ij}} \cdot g_1^{r_j}, B_1^{\epsilon a_{ij}} \cdot spk^{r_j})\}_{j=1}^{K_i} \wedge \{C'_k = (C'_{k0}, C'_{k1}) = (C_{k0}^{-\epsilon}, C_{k1}^{-\epsilon})\}_{k=1}^{K'} \wedge \\ & \{V'_k = (V'_{k0}, V'_{k0}) = (\Psi(V_{k0} \cdot g_1^{\theta_k}), \Psi(V_{k1} \cdot spk^{\theta_k})\}_{k=1}^{K'} \wedge \\ & D^{UID_i} Q^{\omega} = \hat{g} \wedge |e| < 2^{\tau_1 + \tau_2 + l + 2} \wedge \mathsf{Com} = g^{UID_i} h^r \wedge \mathsf{Com}' = \hat{g}^{UID_i} \hat{h}^{\hat{r}} \wedge UID_i \in [2^{l-1}, 2^l]\}.\end{aligned}$$

Formula 1: Proof of the on-chain authentication request

W_i is the aggregated result of $attr'_i$ by aggregating U_{ij} from $j = 1$ to K_i where

$$\begin{aligned} W_i = (W_{i0}, W_{i1}) &= (\prod_{j=1}^{K_i} U_{ij0}, \prod_{j=1}^{K_i} U_{ij1}) \\ &= (g_1^{\sum_{j=1}^{K_i} (\beta \epsilon a_{ij} + r_j)}, g_1^{\epsilon \gamma \sum_{j=1}^{K_i} a_{ij}} spk^{\sum_{j=1}^{K_i} (\epsilon \beta a_{ij} + r_j)}); \end{aligned}$$

$\mathbb{C}'$ is re-randomized and encrypted authentication policy, the re-randomization operation is to ensure that it contains same randomness ϵ as W_i, where

$$C'_k = (C'_{k0}, C'_{k1}) = (C_{k0}^{-\epsilon}, C_{k1}^{-\epsilon}) = (g_1^{-\epsilon \alpha_k}, spk^{-\epsilon \alpha_k} \cdot g_1^{-\epsilon \gamma \Xi_k}).$$

According to the homomorphic property, encrypted identity attributes and authentication policies can be matched without decryption as $\mathbb{V}$, where

$$\begin{aligned} V_{k0} &= g_1^{\sum_{j=1}^{K_i} (\beta \epsilon a_{ij} + r_j) - \epsilon \alpha_k}; \\ V_{k1} &= g_1^{\epsilon \gamma \sum_{j=1}^{K_i} a_{ij} - \epsilon \gamma \Xi_k} spk^{\sum_{j=1}^{K_i} (\epsilon \beta a_{ij} + r_j) - \epsilon \alpha_k}. \end{aligned}$$

There are total K' matching results. If the holder's identity attributes match the authentication policy in the ciphertext format, at least one corresponding matching result (plaintext) is 0, i.e., $\epsilon \gamma \sum_{j=1}^{K_i} a_{ij} - \epsilon \gamma \Xi_k = 0$ (the plaintext corresponding to ciphertext V_k is $g_1^0 = 1^{\mathbb{G}_1}$); however, if $\mathbb{V}$ is straightforwardly uploaded to the blockchain platform, the authentication event reveals the user's authentication pattern, i.e., which authentication policy matches the user's attributes. Malicious adversaries may count the number of times each authentication policy is matched and accordingly infer more private information from the statistics based on some background information. Hence, we apply the permutation and randomization algorithm to shuffle the ciphertext such that the identity committee members cannot know the authentication pattern in step-15, and obtain the final encrypted matching result $\mathbb{V}' = (V'_1, V'_2, ..., V'_{K'})$ using randomness θ_k as

$$V'_k = (V'_{k0}, V'_{k1}) = (\Psi(V_{k0} \cdot g_1^{\theta_k}), \Psi(V_{k1} \cdot spk^{\theta_k})).$$

Step-16, step-17, step-18, and step-20 generate some intermediate results based on the RSA accumulator scheme; step-21 generates a composite zero-knowledge proof which shows that the authentication request, auth, is correctly formed, as shown in Formula 1. The proof verifies five claims of the holder: 1) the holder owns a valid identity credential, $Cred_i$, and attributes, $attr_i$, that are issued by valid identity issuers; 2) the credential is not revoked by valid identity issuers; 3) the attributes are embedded into Elgamal ciphertext, $attr'_i$; 4) the matching algorithm that matches the holder's attributes with the verifier's authentication policy is correctly performed in ciphertext, i.e., $\mathbb{C}'$, $\mathbb{V}'$ and intermediate results are correctly generated; and 5) the real identity of the holder can be traced by the identity auditor. Due to the zero-knowledge property provided by the zero-knowledge proof, the holder can prove the claims without sacrificing the privacy; step-22 generates short-term off-chain access token Δ for off-chain access.

Since each identity committee member monitors the status of the authentication contract, after function *Auth* is triggered by the identity holder and the transaction is confirmed, com-

Algorithm 2 On-chain Authentication Request Generation

```
Input: credential $Cred_i$ and identity attributes $attr_i$;
Output: on-chain authentication request $(aid, \sigma, \mathsf{auth}, ID, \pi_{auth}, \Delta)$;
choosing unique authentication identifier $aid \leftarrow \{0,1\}^{128}$;
choosing $v_i \in Z_p^*$ and $t_i \in Z_p^*$, randomizing $\sigma_i$ as $\sigma_i'$, similar to the computations in sub-phase-1, and $\sigma = \sigma_i'$;
downloading $B$ and $\mathbb{C}$ from the authentication contract;
downloading revocation list $\mathsf{RL} = (UID_1', UID_2', ..., UID_{\mathcal{K}})$ and accumulator $Q = \hat{g}^{\prod_{UID' \in RL} UID'}$ from the revocation contract;
choosing $\epsilon \in Z_p^*$;
for $j = 1$ to $K_i$ do
    encrypting $attr_i = (a_{i1}, a_{i2}, ..., a_{iK_i})$ as $attr_i' = (U_{i1}, U_{i2}, ..., U_{iK_i})$ using $\epsilon$;
end for
aggregating $attr_i'$ to obtain $W_i = (W_{i0}, W_{i1})$;
for $k = 1$ to $K'$ do
    randomizing $\mathbb{C}$ as $\mathbb{C}' = (C_1', C_2', ..., C_{K'}')$ using $\epsilon$;
    calculating $V_k = (V_{k0}, V_{k1}) = (W_{i0} \cdot C_{k0}', W_{i1} \cdot C_{k1}')$;
    choosing $\theta_k \in Z_p^*$;
end for
choosing random permutation $\Psi()$ and shuffling $\mathbb{V} = (V_1, V_2, ..., V_{K'})$ as $\mathbb{V}' = (V_1', V_2', ..., V_{K'}')$ using $\theta_k$;
calculating $h = H_1(aid||B||ts)$ where $ts$ is the current timestamp;
choosing $r \in Z_p^*$ and $\hat{r} \in [1, \frac{N}{2}]$, and generating $\mathsf{Com} = g^{UID_i} h^r$ and $\mathsf{Com}' = \hat{g}^{UID_i} \hat{h}^{\hat{r}}$;
calculating $\mu \cdot UID_i + \omega \cdot \prod_{UID' \in \mathsf{RL}} UID' = 1$ and witness $D = \hat{g}^{\mu}$;
$\mathsf{auth} = (attr_i', \mathbb{C}', \mathbb{V}')$;
choosing random number $\tilde{r} \in Z_p^*$, and generating $ID = (ID_1 = g^{\tilde{r}}, ID_2 = g^{UID_i} \tilde{pk}^{\tilde{r}})$;
generating non-interactive zero-knowledge proof of knowledge $\pi_{auth}$ as shown in Formula 1;
choosing a random number $\delta \in Z_p^*$ and calculates $\Delta = g^{\delta}$;
Output $(aid, \sigma, \mathsf{auth}, ID, \pi_{auth}, \Delta)$;
```

mittee member, ICM_i, downloads authentication identifier aid and authentication content auth, and executes Algorithm 3 to recover the final authentication result based on the Lagrange interpolation method. In step-3, additional randomness $H_2(spk, ts)$ is introduced into the final authentication result where ts is the timestamp of the authentication request. The randomness is to prevent malicious identity committee members from extracting the hidden authentication policy even if they collude with malicious users. In step-5, the proof verifies that ICM_i indeed submits correct secret shares that can be utilized for checking the final authentication result,

$$\begin{aligned}\pi_{ss_i} \leftarrow & ZkPoK\{(sk_i) : aux_i = g^{sk_i} \\ & \wedge \{T_{ik} = e(V_{k0}', H_2(spk||ts))^{sk_i}\}_{k=1}^{K'}\}.\end{aligned}$$

If T committee members have submitted their votes, the state of authentication contract changes. The holder can check the state of on-chain authentication. If $State[aid] = 2$, the on-chain authentication passes and the identity holder stores (aid, Δ) as an off-chain authentication token; otherwise, the authentication fails. If a malicious committee member does not submit their vote according to the protocol, it can be easily identified by the identity verifier via checking the state of function *Vote* in the authentication contract.

This simple case can be extended to support multiple identity credentials $(Cred_1, Cred_2, ...)$ and flexible cross-domain authentication without any restriction. Namely, the identity attribute categories do not need to be confined in a small domain defined by one identity issuer II_i.

In sub-phase-3, an identity holder who passed the on-chain authentication, can authenticate itself to the identity verifier off-chain efficiently many times. The identity holder generates non-interactive zero-knowledge proof of knowledge π_{Δ} with current timestamp ts, and sends $(aid, \Delta, \pi_{\Delta})$ as an off-chain authentication request to the identity verifier. The proof can be generated as:

$$\pi_{\Delta} \leftarrow ZkPoK\{(\delta) : \Delta = g^{\delta}\}.$$

After receiving the request, the identity verifier triggers function *Query* to obtain $Rst[aid]$ and $TmpCreds[aid]$ from the authentication contract. Then, it checks whether $Rst[aid] = True$ and $TmpCreds[aid] = \Delta$. Also, it checks the validity of proof π_{Δ}. If the proof is valid, it means that the holder knows secret δ in Δ stored in the blockchain. If all equations hold and the proof is valid, the authentication request is accepted, and the verifier can negotiate a session symmetric key with the identity holder, and allow the holder to access its resources according to its permissions defined in the authentication contract. Moreover, the verifier also records this off-chain authentication and access attempt by sending transaction (aid, ts, info) to the contract to trigger function *Record*, where info includes the description information of this access attempts, e.g., what kinds of resources the holder accesses. Furthermore, the verifier can blacklist short-term authentication token Δ anytime if necessary. The verifier knows the time that the token has been used by querying the authentication contract.

Algorithm 3 On-chain Authentication Verification

```
Input: authentication identifier $aid$, authentication content $\mathsf{auth}$, and proof $\pi_{auth}$;
Output: a voting transaction $(aid, i, ss_i, \pi_{ss_i}, 1)$ or $(aid, i, null, null, 0)$;
1:  if proof $\pi_{auth}$ holds then
2:      for $k = 1$ to $K'$ do
3:          computing $T_{ik} = e(V_{k0}', H_2(spk||ts))^{sk_i}$;
4:      end for
5:      generating the secret share $ss_i = (T_{i1}, T_{i2}, ..., T_{iK'})$;
6:      generating non-interactive zero-knowledge proof of knowledge $\pi_{ss_i}$;
7:      sending the transaction $(aid, i, ss_i, \pi_{ss_i}, 1)$ to trigger function Vote;
8:  else
9:      sending the transaction $(aid, i, null, null, 0)$ to trigger function Vote;
10: end if
```

Identity Auditing: If an authentication event needs to be traced, the identity auditor can download encrypted identifier ID_i based on authentication identifier aid by triggering function *Audit* in the authentication contract. The auditor decrypts ID to obtain $g^{UID_i} = ID_2 \cdot ID_1^{-\tilde{sk}}$. The identity auditor can inform identity issuer II_i to match g^{UID_i} with RID_i in its local database. If $RID_i = g^{UID_i}$, the identity auditor can trace the unique identifier to identify the holder and link the holder's all authentication events.

Credential Revocation: An identity issuer can revoke previous issued identity credentials of its users. Note that, a revocation contract is designed and deployed by identity committee members in the blockchain platform that contains all revocation information of identity issuers. The pseudocode of the revocation contract is shown in Contract 2. The contract should record different identity issuer's revocation information independently considering that each identity issuer can revoke only their issued credentials. The issuer is allowed to accumulate the revocation information into one accumulated value such that the computational overhead and the storage cost can be reduced. In this setting, there may exist malicious identity issuers that revoke identity credentials not belong to them. To address such an attack, we allow identity issuer II_i to revoke only unique identifier $UID_i = H(pk_i||ts||ind)$. Based on the collision resistance of the hash function, it is impossible for malicious identity issuers to launch such an attack.

The revocation contract consists of four functions: *Init*, *Join*, *Revoke*, and *Query*. Function *Init* is triggered by the identity committee to initialize the parameters used for credential revocation, including revocation list RL, detailed revocation information RvkInfo, accumulated revoked value Q, and the set of registered identity issuers. Functions *Join* and *Revoke* are triggered by the identity issuers to register themselves on the revocation contract and revoke invalid identity credentials. Function *Query* can be triggered publicly by anyone for downloading the latest revocation list and accumulated value.

```
  Init:   Set RL := {} and RvkInfo = {};
          Set $Q = \hat{g}$ and $IIs = \{\}$;
  Join:   Upon receiving from $II_i()$
              Assert $0 = IIs.isExist(II_i)$;
              $IIs.push(II_i)$;
Revoke:   Upon receiving from $II_i(\mathsf{rvkinfo}, RL')$
              Assert $1 = IIs.isExist(II_i)$;
              Assert $1 = RL'.isFormat()$;
              Assert $\emptyset = RL \cap RL'$;
              RvkInfo.$push(II_i, RL', \mathsf{rvkinfo})$;
              $Q = Q^{\prod_{UID' \in RL'} UID'}$;
              $RL = RL \cap RL'$;
 Query:   Upon receiving from ()
              return $(Q, RL)$;
```

Contract 2: Pseudocode of the revocation contract

V. Security Analysis

The simulation-based approach is adopted to capture the security and privacy notions defined in the adversarial model [33]. We define an ideal world assuming that a trusted party $\mathcal{TP}$ exists to execute the necessary authorization and authentication operations, and the real-world is the protocols presented in the proposed framework. We prove that these protocols are secure if we can simulate an ideal world that cannot be distinguished from the real world by the adversaries. In other words, the adversaries cannot obtain any useful information from the proposed system, and the system is secure by default.

Consider a static adversary model in which the numbers of adversaries are fixed during the system setup. We use $\mathcal{U}_{HI}$, $\mathcal{U}_{HH}, \mathcal{U}_{HV}, \mathcal{U}_{HC}$ to denote the set of honest identity issuers, holders, verifiers and committee members, and $\mathcal{U}_{AI}$, $\mathcal{U}_{AH}$, $\mathcal{U}_{AV}, \mathcal{U}_{AC}$ to denote the set of dishonest identity issuers, holders, verifiers, and committee members. The dishonest parties are controlled by a probabilistic polynomial-time adversary (PPT), $\mathcal{A}$. In the real world, different parties communicate via the cryptographic protocols while in the ideal world the parties communicate through $\mathcal{TP}$. It is straightforward to design an ideal functionality F_{auth} that achieves secure cross-domain authorization and authentication with the help of $\mathcal{TP}$. Function F_{auth} can realize all security and privacy properties, i.e., anonymity, non-frameability, authenticity, non-repudiation, and replay-resistance. With the assistance of $\mathcal{TP}$ in the ideal world. A leakage function, $\mathcal{L}$, is defined to represent the privacy leakage in the ideal world. According to our setting, the matching result between a holder's attributes and an authentication policy list is leaked even if in the ideal world. In other words, the information that a user is authenticated successfully or not needs to be leaked to achieve the functionality of authentication. Also, the security analysis is performed in a hybrid model where some ideal secure functions, e.g., zero-knowledge proof, can be invoked in the real world for easy understanding.

To prove the security of the proposed protocols, a simulator, $\mathcal{S}$ (a.k.a the adversary in the ideal world), is constructed, which is able to simulate the view of the adversary in the real world. Note that dishonest parties under the control of A can deviate from the cryptographic protocols, and our security analysis is to demonstrate that the view of an adversary is indistinguishable in the simulation and a real execution of the proposed protocols.

Claim 1. *The proposed framework securely realizes F_{auth} in the hybrid model, provided that 1) the PS assumption is hard; 2) the strong RSA assumption is hard; 3) the Elgamal encryption is computationally indistinguishable under chosen-plaintext attack; 4) the Fujisaki-okamoto commitment is computationally binding and statistically hiding; 5) the Pedersen commitment is computationally binding and statistically hiding; 6) the hash function H is collision-resistant; 7) $ZkPoK$ is a simulation-sound zero-knowledge proof of knowledge.*

The proof is divided into two cases according to the parties controlled by $\mathcal{A}$. In case 1, identity holders, issuers, verifiers, and committee members are controlled by $\mathcal{A}$. In case 2, a proper subset of identity holders, issuers, and committee members are controlled by $\mathcal{A}$. The first case covers user anonymity, non-frameability, and replay-resistance properties while the second case covers confidentiality of authentication policy, authenticity, and non-repudiation properties. We ignore the simulations of system setup and the service entity registration phases. Most of the operations in these protocols do not need interactions with other entities and thus will not leak any private information, except for the distributed key generation protocol and RSA parameter generation protocol. As the distributed key generation protocol and RSA parameter generation protocol have been proven that they can be perfectly simulated [36], [37], we omit them here.

Case-1. For PPT adversary $\mathcal{A}$ controlling a subset of identity holders, issuers, verifiers, and committee members in the real world, there exists an ideal world simulator, $\mathcal{S}$, that can simulate the view of $\mathcal{A}$ which is indistinguishable from a real execution of the proposed protocols.

Proof Sketch. In the user registration phase, $\mathcal{A}$ controls identity issuer $II_i \in \mathcal{U}_{AI}$, while $\mathcal{S}$ simulates as identity holder IH_i to II_i in the real world. If $IH_i \in \mathcal{U}_{HH}$, after receiving the registration request from $\mathcal{TP}$ in the ideal world, $\mathcal{S}$ initiates a registration request with $\mathcal{A}$ and utilizes the zero knowledge simulator to simulate proof π_{Com_i}. If $\mathcal{S}$ obtains a valid identity credential from $\mathcal{A}$, it replies "accept" to $\mathcal{TP}$; otherwise, it replies "reject" to $\mathcal{TP}$.

In the authentication policy generation phase and the on-chain authentication phase, $\mathcal{A}$ controls identity verifier $IV_i \in \mathcal{U}_{AV}$ and identity committee member $ICM_j \in \mathcal{U}_{AC}$, while $\mathcal{S}$ simulates as identity holder $IH_i \in \mathcal{U}_{HH}$ and $ICM_j \in \mathcal{U}_{HC}$ to IV_W and $ICM_j \in \mathcal{U}_{AC}$ in the real world. In the real world, proof π_B uploaded by $\mathcal{A}$ can be extracted, provided by the simulation extractability property of $ZkPoK$. If valid witnesses cannot be constructed from the proof, then S sends "error" to $\mathcal{TP}$ as IV_i in the ideal world. After receiving the on-chain authentication request from an anonymous user in the ideal world, if $\mathcal{TP}$ indicates the user satisfies the authentication policy, S can extract the aggregation of attributes Ξ_k for $k = 1, 2, ..., K'$ from proof π_B, select one authentication policy Ξ, simulate the authentication request V'_k based on Ξ, and use the zero-knowledge simulator to simulate proof π_{auth}. Secret share T_{jk} of $ICM_j \in \mathcal{U}_{HC}$ can be simulated since $\mathcal{S}$ generates secret key sk_j, and corresponding proof $\pi_{T_{jk}}$ can be simulated by $\mathcal{S}$ using the zero-knowledge simulator. According to the vote of $ICM_j \in \mathcal{U}_{AC}$ controlled by $\mathcal{A}$, $\mathcal{S}$ votes for the authentication request in the ideal world. In the off-chain authentication phase, after receiving the off-chain authentication request from an anonymous user in the ideal world, $\mathcal{S}$ can use the zero-knowledge simulator to simulate proof π_Δ if the number of votes is equal to T.

From the analysis, we can see that the simulation between $\mathcal{S}$ and $\mathcal{A}$ is perfect. Also, the inputs and outputs of $\mathcal{S}$ in the ideal world are the same as those of $\mathcal{A}$ in the real world.

Case-2. For PPT adversary $\mathcal{A}$ controlling a subset of identity holders, issuers, and committee members in the real world, there exists an ideal world simulator, $\mathcal{S}$, that can simulate the view of $\mathcal{A}$ which is indistinguishable from a real execution of the proposed protocols.

Proof Sketch. In the user registration phase, $\mathcal{A}$ controls identity holder $IH_i \in \mathcal{U}_{AH}$. $\mathcal{S}$ simulates as identity issuer II_i to IH_i in the real world. After receiving the registration request from $\mathcal{A}$ in the real world, $\mathcal{S}$ can extract (u_i, t_i) from proof π_{Com_i}. If the extraction fails, $\mathcal{S}$ aborts; otherwise, $\mathcal{S}$ sends $(i, a_{i1}, a_{i2}, ..., a_{iK_i})$ to $\mathcal{TP}$. If $\mathcal{TP}$ returns "accept", $\mathcal{S}$ issues a valid identity credential to $\mathcal{A}$ following the protocol; otherwise, it rejects the registration request. $\mathcal{S}$ can obtain the valid credential $Cred_i$.

In the on-chain authentication phase, $\mathcal{A}$ controls identity holder $IH_i \in \mathcal{U}_{AH}$ and identity committee member $ICM_j \in \mathcal{U}_{AC}$. $\mathcal{S}$ simulates as identity verifier $IV_i \in \mathcal{U}_{HV}$ and $ICM_j \in \mathcal{U}_{HC}$ to II_i and $ICM_j \in \mathcal{U}_{AC}$ in the real world. Here, $\mathcal{S}$ can simulate base B and hidden authentication policy $\mathbb{C}$ in the authentication contract by choosing random authentication policies. Adversary $\mathcal{A}$ cannot distinguish B and $\mathbb{C}$ due to the indistinguishability of the Elgamal ciphertext. When $\mathcal{A}$ submits an authentication request to the contract, $\mathcal{S}$ can extract attributes $(a_{i1}, a_{i2}, ..., a_{iK_i})$ based on the simulation extractability property of $ZkPoK$. In the ideal world, $\mathcal{S}$ submits $\sum_{j=1}^{K_i} a_{ij}$ to $\mathcal{TP}$, and obtains the authentication result "accept" or "reject". Based on the authentication result, S can set $\zeta_j = \prod_{i=1, i\neq j}^{T} \frac{i}{i-j}$, and simulate the share of honest identity committee member ICM_i given by

$$T_{ik} = \left(\frac{T_k}{\prod_{ICM_j \in \mathcal{U}_{AC}} (T_{jk})^{\zeta_j}} \right)^{\zeta_i^{-1}}.$$

By changing T_k in the above equation, $\mathcal{S}$ can control the on-chain authentication output in the real world and make it indistinguishable from the output of the ideal world, i.e.,

$$(T_{ik})^{\zeta_i} \cdot \prod_{ICM_j \in \mathcal{U}_{AC}} (T_{jk})^{\zeta_j} = T_k.$$

In the off-chain authentication phase, $\mathcal{S}$ can extract δ from the authentication request of $\mathcal{A}$, and submit δ in the ideal world to $\mathcal{TP}$. According to the result received from $\mathcal{TP}$, $\mathcal{S}$ can respond to $\mathcal{A}$.

From the analysis, we can see that the simulation between $\mathcal{S}$ and $\mathcal{A}$ is perfect. Also, the inputs and outputs of $\mathcal{S}$ in the ideal world are the same as those of $\mathcal{A}$ in the real world.

VI. Performance Evaluation

In this section, we evaluate the performance of the proposed framework in terms of computational cost and communication overhead. We have developed a prototype and implemented all the stages and the protocols proposed in the framework using Java language and the MIRACL library (java) [2]. The off-chain performance is evaluated in a laptop with i5-7300U CPU 2.60GHz and 16GB RAM. We use the Java language because it can be supported on a consortium blockchain platform, Hyperledger Fabric, which is the platform used in our on-chain performance evaluation. We deploy a testnet of the Hyperledger fabric on the laptop for the on-chain performance evaluation. In the following, each experimental result is an average experimental result with 20 trials.

We first present the experiment setting, including the security and other performance parameters. The security of our proposed framework is determined by four parameters, security parameter τ of the bilinear group, security parameter $\hat{\tau}$ of the RSA group, and two security parameters (τ_1, τ_2) of

[2] https://github.com/miracl/core

TABLE II: Description of experimental parameters

Parameter	Description
N	number of identity issuers involved
K_i	number of identity attributes issued by II_i
K'	number of authentication polices published by IV
T	number of required identity committee members
$\mathcal{K}$	number of revoked users
l	maximum size of the revocation list

zero-knowledge proof. To guarantee the security requirements, we choose the BN254 pairing-friendly curve to construct our bilinear group where $\tau = 254$. We set $\hat{\tau} = 2048$ to guarantee the security level of the RSA group, and select $\tau_1 = 80$ and $\tau_2 = 80$ in our experiments. We choose 80 bits to guarantee that a malicious user may cheat successfully in the zero-knowledge proof with error probability 2^{-80}, which is negligible. To get a comprehensive performance analysis, several performance parameters are chosen in the simulation as shown in Table II. Moreover, some protocols proposed in the framework are instantiated using existing instantiations, e.g., we use the instantiations in [40], [41] to achieve the range proof and the RSA accumulator proof defined in Formula 1.

A. *Off-chain Computational Cost*

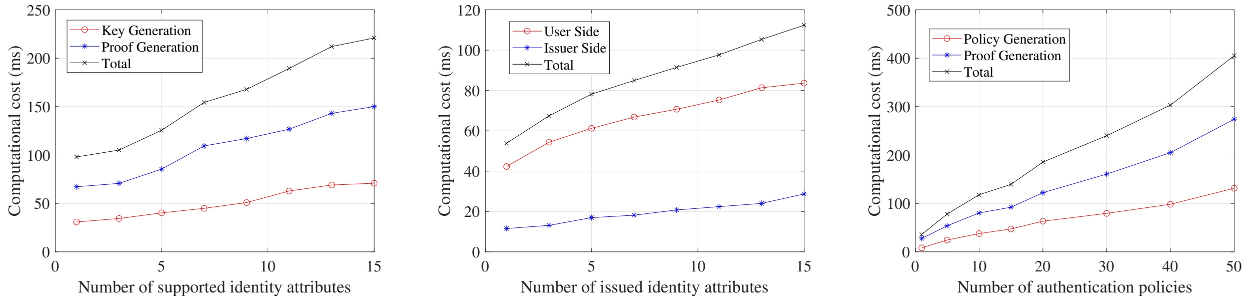
We evaluate the off-chain computational cost of each entity in five stages of the proposed framework, except the system setup stage, in terms of the CPU computational delay. In the service entity registration stage, each identity issuer needs to register only once, and the computational cost is dependent on setting of K_i, and thus K_i varies in our experiment. According to our experiment results, when the number of identity attributes reaches 15, it takes less than 250 ms for a service entity to complete the registration, as shown in Figure 5 (a). In the user registration and credential authorization stage, both users and identity issuers are involved, and the computational efficiency is shown in Figure 5 (b). The computational cost is around 90 ms at the user side and around 30 ms at the identity issuer side when the number of issued attributes reaches 15. In the authentication policy generation stage, each identity verifier needs to generate the authentication policy list and the hidden authentication policies. The computational performance is determined by parameter K'. When K' is large, the identity verifier has to spend more time as shown in Figure 5 (b). When the number of authentication policies reaches 50, the computational delay of the identity verifier is still smaller than 410 ms, which is efficient.

In the on/off-chain authentication stage, there exist three sub-phases. In sub-phase-1, users only authenticate themselves to access the blockchain platform. This authentication process is fast since users do not need to prove anything but the registration information. The experiment results show that it takes 90 ms to achieve the pre-authentication protocol in sub-phase-1 between a user and an identity committee member. In sub-phase-2, users have to generate authentication requests. The off-chain computation of the request is influenced by three parameters: number N of identity issuers involved, number K_i of identity attributes issued by each identity issuer, and number K' of authentication policies. Among them, N and K_i can be viewed as the same type of parameters since changing either of them will have almost the same influence on the experiment results. Consequently, we set $N = 2, 3, 4$ and $K_i = 3, 4, 5$ to represent a sample of possible cross-domain authentication situations in the real world, where a user uses N issuer's K_i attributes to achieve cross-domain authentication. We also change parameter K' in the experiment to illustrate how changing the number of authentication policies can affect the computational efficiency of the user and the committee member at this stage. In addition, the maximum size of revocation list l is an important factor that may decrease the computational efficiency on the user side. Therefore, we set $l = 50$ and $l = 80$ in our experiment to show the difference. Figure 6 and Figure 7 show the computational costs of users and identity committee members increase with the number of authentication policies. With more than 20 identity attributes issued from 4 different identity issuers, a user needs around 1.2 seconds to generate an on-chain authentication request, and the committee member can verify the request in less than 1.1 seconds. In sub-phase-3, a user can create the off-chain authentication request in less than 5 ms as shown in Figure 8, and the computational cost is much lower compared with existing schemes because most of the operations are performed in sub-phase-2. In the identity auditing stage, the computational cost of an identity auditor is low since it performs only the decryption operations to recover a user's unique identifier.

B. *On-chain Computation*

We have deployed the Hyperledger fabric v2.1 framework to build a consortium blockchain in the laptop. The framework consists of two peer nodes belonging to two organizations, and an ordering node that uses the RAFT consensus protocol to achieve the consensus between the peer nodes. The smart contract (also named chain code in the Hyperledger fabric) is written using Java language. There are two contracts deployed on the blockchain platform: the authentication contract (A-Contract) and the revocation contract (R-Contract). Since there exist many related works on performance of the Hyperledger framework, we mainly focus on the performance of the deployed chaincode, i.e., the performance of triggering each function in the smart contract and the time waited to confirm the submitted transactions. The experiment setting is fixed for the on-chain performance evaluation with $N = 2$, $K_i = 3$, $K' = 5$, $l = 50$, $T = 10$. When an identity issuer submits a revocation request to trigger function *Revoke*, the size of the updated revocation list is limited to 100. From the experimental results shown in Table III, it is observed that most of the on-chain functions can be triggered in less than 5 seconds, and the delay is caused by the consensus protocol running on the blockchain.

Communication Overhead: We evaluate the communication overhead of each entity in different stages. We adopt the same experimental setting as that in the on-chain performance evaluation. From the results shown in Table IV, we observe that the largest communication cost is less than 790 KB.



(a)

(b)

(c)

Figure 5: Off-chain computational cost of the registration stage and the policy generation stage: (a) service registration with different K_i; (b) user registration with different K_i; (c) policy generation with different K'

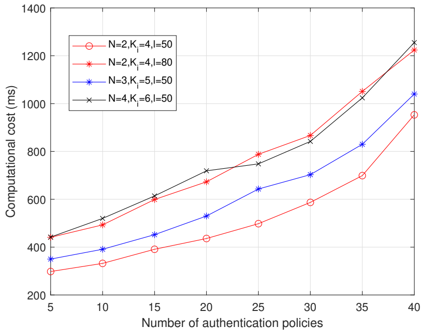


Figure 6: Computational cost of users in sub-phase-2 with different settings

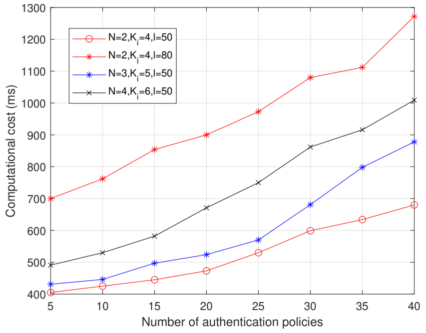


Figure 7: Computational cost of an identity committee member in sub-phase-2 with different settings

TABLE III: Response delay of On-chain transactions

Function	*Init*	*Auth*	*Vote*	*Query*	*Record*	*Audit*
A-Contract	4.8 s	3.4 s	6.5 s	1.7 s	2.5 s	1.9 s
Function	*Init*	*Join*	*Revoke*	*Query*	N/A	N/A
R-Contract	2.8 s	1.5 s	3.6 s	1.6 s	N/A	N/A

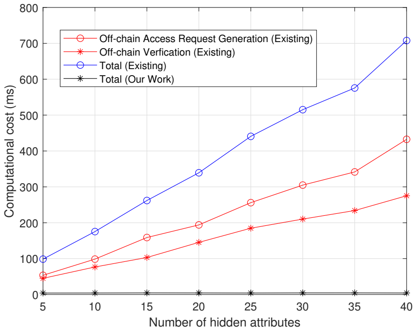


Figure 8: Computational cost in sub-phase-3 with different settings

TABLE IV: Communication overhead

Stages	II	IH	IV	ICM	IA
Service Registration	54 KB	N/A	N/A	N/A	31 KB
User Authorization	22 KB	13 KB	N/A	N/A	N/A
Policy Generation	N/A	N/A	580 KB	N/A	N/A
Authentication	N/A	749 KB	42 KB	783 KB	N/A
Identity Auditing	25 KB	N/A	N/A	N/A	45 KB
Revocation	54 KB	N/A	N/A	N/A	N/A

VII. Conclusion

In this paper, we have proposed a blockchain-assisted cross-domain authorization and authentication framework for smart city. The most significant property of the framework is the authentication transparency, i.e., user authentication events can be audited publicly without violating privacy regulations. Moreover, the framework is suitable for resource-limited devices as the off-chain authentication cost is low, independent of the complexity of the authentication policy. As a generic solution, the proposed framework can be easily adopted to many smart city applications to enhance their security and promote potential collaboration. For the future work, we intend to explore new approaches to compress the on-chain storage cost such that the framework can be more efficient when deploying on a public blockchain.

Acknowledgments

The authors would like to thank Jimmy Qi for the implementations in the consortium blockchain.

References

[1] R. Rodulfo, "Smart city case study: City of coral gables leverages the internet of things to improve quality of life," *IEEE Internet of Things Magazine*, vol. 3, no. 2, pp. 74–81, 2020.

[2] C. Xu, N. Wang, L. Zhu, C. Zhang, K. Sharif, and H. Wu, "Reliable and privacy-preserving top-k disease matching schemes for e-healthcare systems," *IEEE Internet of Things Journal*, to appear.

[3] J. Tang, H. Zhu, R. Lu, X. Lin, H. Li, and F. Wang, "DLP: Achieve customizable location privacy with deceptive dummy techniques in LBS applications," *IEEE Internet of Things Journal*, to appear.

[4] Q. Yang, S. Fu, H. Wang, and H. Fang, "Machine-learning-enabled cooperative perception for connected autonomous vehicles: Challenges and opportunities," *IEEE Network*, vol. 35, no. 3, pp. 96–101, 2021.

[5] Y. Su, Y. Li, J. Li, and K. Zhang, "LCEDA: Lightweight and communication efficient data aggregation scheme for smart grid," *IEEE Internet of Things Journal*, to appear.

[6] M. Li, D. Hu, C. Lal, M. Conti, and Z. Zhang, "Blockchain-enabled secure energy trading with verifiable fairness in industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 10, pp. 6564–6574, 2020.

[7] L. Qi, C. Hu, X. Zhang, M. R. Khosravi, S. Sharma, S. Pang, and T. Wang, "Privacy-aware data fusion and prediction with spatial-temporal context for smart city industrial environment," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 6, pp. 4159–4167, 2020.

[8] P. Kumar, R. Kumar, G. Srivastava, G. P. Gupta, R. Tripathi, T. R. Gadekallu, and N. Xiong, "PPSF: A privacy-preserving and secure framework using blockchain-based machine-learning for IoT-driven smart cities," *IEEE Transactions on Network Science and Engineering*, to appear.

[9] J. Guo, X. Ding, and W. Wu, "A blockchain-enabled ecosystem for distributed electricity trading in smart city," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 2040–2050, 2020.

[10] K. Zhang, J. Ni, K. Yang, X. Liang, J. Ren, and X. Shen, "Security and privacy in smart city applications: Challenges and solutions," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 122–129, 2017.

[11] C. Huang, R. Lu, J. Ni, and X. Shen, "DAPA: A decentralized, accountable, and privacy-preserving architecture for car sharing services," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 4869–4882, 2020.

[12] Y. Zhang, C. Xu, H. Li, K. Yang, N. Cheng, and X. Shen, "Protect: Efficient password-based threshold single-sign-on authentication for mobile users against perpetual leakage," *IEEE Transactions on Mobile Computing*, vol. 20, no. 6, pp. 2297–2312, 2020.

[13] M. Shen, H. Liu, L. Zhu, K. Xu, H. Yu, X. Du, and M. Guizani, "Blockchain-assisted secure device authentication for cross-domain industrial IoT," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 5, pp. 942–954, 2020.

[14] G. Cheng, Y. Chen, S. Deng, H. Gao, and J. Yin, "A blockchain-based mutual authentication scheme for collaborative edge computing," *IEEE Transactions on Computational Social Systems*, to appear.

[15] H. Zhang, X. Chen, X. Lan, H. Jin, and Q. Cao, "BTCAS: A blockchain-based thoroughly cross-domain authentication scheme," *Journal of Information Security and Applications*, vol. 55, p. 102538, 2020.

[16] R. N. Zaeem and K. S. Barber, "The effect of the gdpr on privacy policies: Recent progress and future promise," *ACM Transactions on Management Information Systems*, vol. 12, no. 1, pp. 1–20, 2020.

[17] H. S. G. Pussewalage and V. A. Oleshchuk, "An anonymous delegatable attribute-based credential scheme for a collaborative e-health environment," *ACM Transactions on Internet Technology*, vol. 19, no. 3, pp. 1–22, 2019.

[18] D. Pointcheval and O. Sanders, "Short randomizable signatures," in *Proceedings of CT-RSA*, 2016, pp. 111–126.

[19] S. Gao, G. Piao, J. Zhu, X. Ma, and J. Ma, "TrustAccess: A trustworthy secure ciphertext-policy and attribute hiding access control scheme based on blockchain," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5784–5798, 2020.

[20] V. Sucasas, G. Mantas, M. Papaioannou, and J. Rodriguez, "Attribute-based pseudonymity for privacy-preserving authentication in cloud services," *IEEE Transactions on Cloud Computing*, to appear.

[21] J. Li, W. Zhang, V. Dabra, K.-K. R. Choo, S. Kumari, and D. Hogrefe, "AEP-PPA: An anonymous, efficient and provably-secure privacy-preserving authentication protocol for mobile services in smart cities," *Journal of Network and Computer Applications*, vol. 134, pp. 52–61, 2019.

[22] A. Yang, J. Weng, K. Yang, C. Huang, and X. Shen, "Delegating authentication to edge: A decentralized authentication architecture for vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, to appear.

[23] J. Li, M. H. Au, W. Susilo, D. Xie, and K. Ren, "Attribute-based signature and its applications," in *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, 2010, pp. 60–69.

[24] Y. Li, X. Chen, Y. Yin, J. Wan, J. Zhang, L. Kuang, and Z. Dong, "SDABS: A flexible and efficient multi-authority hybrid attribute-based signature scheme in edge environment," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 3, pp. 1892–1906, 2020.

[25] H. Cui, R. H. Deng, and G. Wang, "An attribute-based framework for secure communications in vehicular ad hoc networks," *IEEE/ACM Transactions on Networking*, vol. 27, no. 2, pp. 721–733, 2019.

[26] J. Sun, Y. Su, J. Qin, J. Hu, and J. Ma, "Outsourced decentralized multi-authority attribute based signature and its application in IoT," *IEEE Transactions on Cloud Computing*, to appear.

[27] Y. Chen, J. Li, C. Liu, J. Han, Y. Zhang, and P. Yi, "Efficient attribute based server-aided verification signature," *IEEE Transactions on Services Computing*, to appear.

[28] H. Xiong, Y. Bao, X. Nie, and Y. I. Asoor, "Server-aided attribute-based signature supporting expressive access structures for industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 2, pp. 1013–1023, 2019.

[29] X. Liu, A. Yang, C. Huang, Y. Li, T. Li, and M. Li, "Decentralized anonymous authentication with fair billing for space-ground integrated networks," *IEEE Transactions on Vehicular Technology*, to appear.

[30] X. Jia, N. Hu, S. Su, S. Yin, Y. Zhao, X. Cheng, and C. Zhang, "IRBA: an identity-based cross-domain authentication scheme for the internet of things," *Electronics*, vol. 9, no. 4, p. 634, 2020.

[31] J. Chen, Z. Zhan, K. He, R. Du, D. Wang, and F. Liu, "XAUTH: Efficient privacy-preserving cross-domain authentication," *IEEE Transactions on Dependable and Secure Computing*, 2021.

[32] G. Ali, N. Ahmad, Y. Cao, S. Khan, H. Cruickshank, E. A. Qazi, and A. Ali, "XDBAUTH: Blockchain based cross domain authentication and authorization framework for internet of things," *IEEE Access*, vol. 8, pp. 58 800–58 816, 2020.

[33] Y. Lindell, "How to simulate it–a tutorial on the simulation proof technique," *Tutorials on the Foundations of Cryptography*, pp. 277–346, 2017.

[34] J. Camenisch, M. Drijvers, A. Lehmann, G. Neven, and P. Towa, "Short threshold dynamic group signatures," in *Proceedings of SCN*, 2020, pp. 401–423.

[35] Y. Liu, D. He, M. S. Obaidat, N. Kumar, M. K. Khan, and K.-K. R. Choo, "Blockchain-based identity management systems: A review," *Journal of network and computer applications*, vol. 166, p. 102731, 2020.

[36] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Secure distributed key generation for discrete-log based cryptosystems," *Journal of Cryptology*, vol. 20, no. 1, pp. 51–83, 2007.

[37] T. K. Frederiksen, Y. Lindell, V. Osheter, and B. Pinkas, "Fast distributed rsa key generation for semi-honest and malicious adversaries," in *Proceedings of Crypto*, 2018, pp. 331–361.

[38] E. Androulaki *et al.*, "Hyperledger Fabric: A distributed operating system for permissioned blockchains," in *Proceedings of EuroSys*, 2018, pp. 1–15.

[39] D. Maram, H. Malvai, F. Zhang, N. Jean-Louis, A. Frolov, T. Kell, T. Lobban, C. Moy, A. Juels, and A. Miller, "CanDID: Can-do decentralized identity with legacy compatibility, sybil-resistance, and accountability," in *Proceedings of IEEE S&P*, 2021, pp. 645–663.

[40] D. Benarroch, M. Campanelli, D. Fiore, and D. Kolonelos, "Zero-knowledge proofs for set membership: Efficient, succinct, modular." *IACR ePrint*, vol. 2019, p. 1255, 2019.

[41] J. Camenisch, R. Chaabouni *et al.*, "Efficient protocols for set membership and range proofs," in *IACR Asiacrypt*, 2008, pp. 234–252.

Cheng Huang (Member, IEEE) received his B.Eng and M.Eng in information security from Xidian University, China, in 2013 and 2016 respectively, and received the Ph.D. degree in Electrical and Computer Engineering, University of Waterloo, ON, Canada in 2020. He is currently a Postdoctoral Research Fellow with the Department of Electrical and Computer Engineering, University of Waterloo. His research interests are in the areas of applied cryptography, cyber security and privacy in the mobile network.

Liang Xue (Student Member, IEEE) received her B.S. and M.S. degree in School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC), China in 2015 and 2018, respectively. Currently, She is pursuing the PhD degree at the Department of Electrical and Computer Engineering, University of Waterloo, Canada. Her research interests include applied cryptography, cloud computing, and blockchain.

Dongxiao Liu (Member, IEEE) is a Postdoctoral Research Fellow with the Department of Electrical and Computer Engineering, University of Waterloo. He received the PhD degree at the Department of Electrical and Computer Engineering, University of Waterloo, Canada in 2020. His research interests include security and privacy in intelligent transportation systems, blockchain, and mobile networks.

Xuemin (Sherman) Shen (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990. He is a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research focuses on network resource management, wireless network security, Internet of Things, 5G and beyond, and vehicular ad hoc and sensor networks. Dr. Shen is a registered Professional Engineer of Ontario, Canada, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, a Chinese Academy of Engineering Foreign Member, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society.

Dr. Shen received the Canadian Award for Telecommunications Research from the Canadian Society of Information Theory (CSIT) in 2021, the R.A. Fessenden Award in 2019 from IEEE, Canada, Award of Merit from the Federation of Chinese Canadian Professionals (Ontario) in 2019, James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, Joseph LoCicero Award in 2015 and Education Award in 2017 from the IEEE Communications Society, and Technical Recognition Award from Wireless Communications Technical Committee (2019) and AHSN Technical Committee (2013). He has also received the Excellent Graduate Supervision Award in 2006 from the University of Waterloo and the Premier's Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada. He served as the Technical Program Committee Chair/Co-Chair for IEEE Globecom' 16, IEEE Infocom'14, IEEE VTC'10 Fall, IEEE Globecom'07, and the Chair for the IEEE Communications Society Technical Committee on Wireless Communications. Dr. Shen is the President of the IEEE Communications Society. He was the Vice President for Technical & Educational Activities, Vice President for Publications, Member-at-Large on the Board of Governors, Chair of the Distinguished Lecturer Selection Committee, Member of IEEE Fellow Selection Committee of the ComSoc. Dr. Shen served as the Editor-in-Chief of the IEEE IoT JOURNAL, IEEE Network, and IET Communications.

Weihua Zhuang (Fellow, IEEE) received the PhD degree in electrical engineering from the University of New Brunswick, Canada. She has been with the Department of Electrical and Computer Engineering, University of Waterloo, Canada, since 1993, where she is a University Professor and a Tier I Canada Research Chair in Wireless Communication Networks. Dr. Zhuang is the recipient of 2021 Women's Distinguished Career Award from IEEE Vehicular Technology Society, 2021 Technical Contribution Award in Cognitive Networks from IEEE Communications Society, and 2021 R.A. Fessenden Award from IEEE Canada. She was the Editor-in-Chief of the IEEE Transactions on Vehicular Technology from 2007 to 2013, Technical Program Chair/Co-Chair of IEEE VTC 2017/2016 Fall, Technical Program Symposia Chair of IEEE Globecom 2011, and an IEEE Communications Society Distinguished Lecturer from 2008 to 2011. She is an elected member of the Board of Governors and Vice President for Publications of the IEEE Vehicular Technology Society. Dr. Zhuang is a Fellow of the Royal Society of Canada, Canadian Academy of Engineering, and Engineering Institute of Canada.

Rob Sun is currently a principal engineer with Huawei Technologies Canada Co. Ltd. His work primarily focuses on the advancement of NG wireless, including 5G/6G and WiFi/IoT security architecture and standardization. He was also Vice Chair of the IEEE Privacy Management Protection Task Group, which was to set out the best practices for protecting personal privacy information, and support efficient, adaptable, and innovative approaches for privacy governance. He was regarded as one of the core contributors to the standardization of a series of NG WiFi security protocols and certifications, including the most recent WiFi WPA3 protocol suites. He has also co-authored a few books on wireless security technologies.

Bidi Ying is currently working at Huawei Technologies Canada Co., Ltd. as a senior network architecture engineer. Her main research is about security and privacy in wireless networks. Before that, she worked at the University of Ottawa. During the past 15 years, she has published more than 200 papers in top conferences and reputable journals.