Privacy-Preserving Educational Credentials Management Based on Decentralized Identity and Zero-Knowledge Proof

Tianmin Xiong, Zhao Zhang(✉), and Cheqin Jing

School of Data Science and Engineering,
East China Normal University, Shanghai, China
tianminxiong@stu.ecnu.edu.cn, {zhzhang,cqjin}@dase.ecnu.edu.cn

Abstract. Educational credentials are an important part of the education ecosystem. However, these credentials often exist in the form of traditional paper documents, which are issued slowly, susceptible to forgery, and entail high verification costs. Moreover, electronic educational credentials stored on centralized servers may not adequately ensure their authenticity and can potentially expose sensitive student information. Historical educational credentials lacked a standardized formats, impeding efficient sharing and verification between parties. Additionally, conventional encryption schemes only safeguarded credential privacy during storage, posing challenges in protecting privacy while presenting the credentials. We propose a privacy-preserving educational credential management scheme based on decentralized identity and zero-knowledge proof. The scheme incorporates a decentralized identity authentication model utilizing blockchain technology, enabling flexible issuance and point-to-point sharing of educational credentials. Furthermore, by employing privacy-preserving credentials designed with zero-knowledge proofs, we can eliminate potential privacy breaches when students present their credentials. We also have developed a system prototype and conducted performance evaluations, thereby validating the correctness and practicality of our scheme.

Keywords: Educational Credential · Privacy Preservation · Decentralized Identity · Zero-Knowledge Proof · Blockchain

1 Introduction

In the field of education, students constitute the largest group and possess numerous educational credentials, including transcripts, recommendation letters, and various credentials such as diplomas, degrees, internships, and training records. These credentials are typically issued by universities, educational institutions, or authoritative institutions, carrying legal validity and reputation assurance. The secure sharing of these educational credentials is essential for the functioning of the education ecosystem and for facilitating the recruitment process in companies.

W. Hong and G. Kanaparan (Eds.): ICCSE 2023, CCIS 2023, pp. 243–255, 2024.
https://doi.org/10.1007/978-981-97-0730-0_22

Currently, many educational institutions employ centralized servers to store electronic education credentials. However, these credentials are often based on paper documents, resulting in slow issuance times, high verification costs. Moreover, traditional centralized server systems are vulnerable to credential tampering, thereby compromising the authenticity of student credentials. Additionally, there is also a risk of sensitive information leakage such as student birthdays, ID numbers, and course scores. For instance, during the scholarship review process, it is necessary for the committee to manually verify each student's course scores, credentials, and honors against their information in the student information system, which could lead to unauthorized access to their confidential details.

Blockchain is a distributed database that is decentralized, tamper-proof, traceable, and jointly maintained [1]. Researchers have leveraged this technology to develop robust, transparent, and trustworthy educational sharing platforms. For instance, studies [2–4] have utilized blockchain technology to create secure and tamper-proof platforms for students to store and share their educational credentials with external institutions. Additionally, recording students' course credit records on the blockchain enables credit sharing between different educational institutions, enhancing educational mobility for students [3,5]. However, while blockchain ensures the integrity and authenticity of educational credentials, it remains susceptible to sensitive information leakage due to its public and reviewable nature. To address this issue, some studies [6,7] suggest encrypting credentials, but delegating identity and keys to a single issuer lacks the ability to achieve autonomous security.

Moreover, when presenting credentials, sharing complete plain-text credentials can still expose students' sensitive information. Although verification schemes based on zero-knowledge proofs facilitate the validation of predicate proofs concerning credentials, they do not offer a means to authenticate the origin of the credential. Furthermore, users can easily forge credentials that meet the verification conditions. To address these limitations, zk-creds [8] introduced the LinkG16 protocol, which employs blinding and linking techniques to combine multiple zero-knowledge proofs. This ensures that all individual proofs originate from the same credential.

To address the above challenges, we have designed a privacy-preserving educational credentials management scheme based on decentralized identity and zero-knowledge proof. This scheme introduces a decentralized identity authentication model based on blockchain, allowing educational credentials to be flexibly issued and peer-to-peer shared. Additionally, by employing a privacy-preserving credential design based on zero-knowledge proof, potential privacy breaches during credential presentation can be mitigated. We have designed a concise Merkle Tree to maintain an on-chain issuance list to prove the source of the credentials. Finally, we have implemented a system prototype based on the CITA [9] consortium chain and analyzed its performance to validate the correctness and practicality of the privacy-preserving credentials.

The remainder of this paper is structured as follows. Section 2 provides related work on decentralized identity and zero-knowledge proof. Section 3 explains the proposed system model. Implementation and experimental results are presented in Sect. 4. Finally, Sect. 5 concludes the paper.

2 Relate Work

2.1 Blockchain and Decentralized Identity

Blockchain is a revolutionary technological framework that combines distributed data storage, peer-to-peer (P2P) protocols, consensus mechanisms, asymmetric encryption, and other computer technologies. It serves as a decentralized and tamper-proof database. The stored data remains immutable and transparent, allowing for enhanced traceability and other essential characteristics. Blockchain technology leverages chain data structures for data verification and storage, distributed node consensus algorithms for data generation and updates, cryptographic techniques to ensure secure data transmission and access, and smart contracts comprising automated script codes for data programming and manipulation.

The Merkle tree, a binary tree data structure extensively employed in the blockchain domain, serves as a fundamental method for validating the integrity and authenticity of large-scale data sets. Constructing a Merkle tree involves segmenting the data into fixed-size blocks, which are then hashed to establish the hierarchical tree structure. Each internal node's hash value is computed based on the hashes of its child nodes, while the root node's hash value provides verification for the entire dataset. The Merkle tree's verification path comprises the hash values of all non-leaf nodes traversed from the data block to the root node. By scrutinizing this path, one can ascertain the inclusion of a specific data block within the corresponding Merkle tree and promptly identify any potential data tampering.

Blockchain-based decentralized identity is a solution for decentralized identity authentication and management. It leverages the blockchain's decentralization features and non-tamperability to ensure secure storage, verification, and authorization management of identity information. By doing so, it addresses issues encountered in traditional identity verification methods, such as single points of failure, account switching, and data leakage. The decentralized identity authentication model utilizes smart contracts to execute identity authentication and management processes. Each user possesses a unique decentralized identity identifier(DID) [10], generated through an asymmetric key, ensuring its immutability and uniqueness.

The W3C Credentials Community Group has spearheaded the development of a series of standardized solutions for decentralized identity. These solutions are built upon blockchain-based distributed ledger technology and the DID protocol. The protocol primarily encompasses two key components: the decentralized identity identifier and the Verifiable Credential (VC) [11]. The VC consists of a declaration file containing holder identity attributes and issuer information,

which is presented to the verifier during the authentication process. Figure 1 illustrates the general structure of the DID authentication model, involving three main entities: the Holder, the Issuer, and the Verifier. The Holder is responsible for storing and managing personal identity information along with related verifiable credentials. The Issuer, on the other hand, verifies the holder's identity, manages the schema, issues verifiable credentials based on the schema, and provides trust endorsement. Lastly, the Verifier verifies the holder's identity and examines the verifiable credentials information.

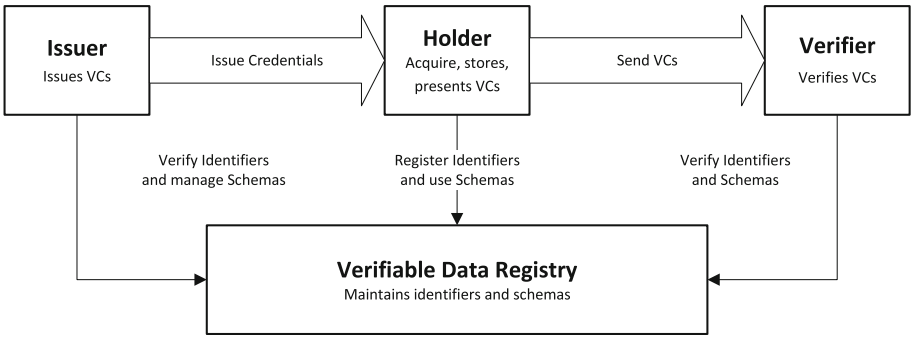

Fig. 1. W3C Decentralized Identity Authentication Model

2.2 Zero-Knowledge Proof

Zero-Knowledge Proof, initially proposed by Goldwasser, Micali, and Rackoff, is a cryptographic protocol that involves two parties: the prover and the verifier [12]. It serves the purpose of proving membership or knowledge in a secure manner. Zero-knowledge proof possesses three fundamental properties:

(1) Completeness: This property verifies the correctness of the protocol itself. If the prover possesses valid evidence for a statement and both parties (prover and verifier) honestly execute the protocol, the prover can convince the verifier that the statement is indeed correct.
(2) Soundness: Soundness protects honest verifiers from being deceived by malicious provers. It ensures that an incorrect statement cannot be proven as true by an untrusted party.
(3) Zero knowledge: The concept of zero knowledge implies that the prover can prove the validity of a statement to the verifier without revealing any additional information apart from its correctness. This property safeguards privacy during the proof process.

Zero-knowledge proof offers trust establishment and privacy protection through these properties, making it highly versatile and applicable in various

domains. It finds application not only in classical cryptography, such as public key encryption, digital signatures, and identity authentication but also in blockchain technology, privacy computing, and other emerging fields. Current mainstream zero-knowledge proof protocols, including Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge(zk-SNARK) [13], Zero-Knowledge Scalable Transparent Arguments of Knowledge(zk-STARK) [14], and Bulletproof [15], have shown practicality in specific scenarios.

3 System Model

3.1 Overview

We propose a privacy-preserving educational credentials management scheme based on decentralized identity and zero-knowledge proof. The scheme enables secure issuance, sharing, and verification of students' educational credentials. The system involves four entities: students, verifiers, educational institutions, and the blockchain. An overview of the system is presented in Fig. 2, with the roles and main processes described below:

- **Educational institution** provide students with the plain-text of educational credentials and construct the commitments of the credentials. They then upload these commitments to the issuance list on the blockchain, that is, creating leaves in the Merkle tree.
- **Student** can register a DID and use the DID to access any educational institution. All educational credentials received by the student will be bound to his DID. In the stage of presenting credentials, students need to construct multiple zero-knowledge proofs individually and link them into *link proof*: 1) *pred proof*: the credential satisfies certain predicate; 2) *memb proof*: the credential is a member of the Merkle tree of the issuance list.
- **Verifier** is any potential user, such as other students, companies, etc. The verifier performs verification on the linked zero-knowledge proof: 1) verifies the *pred proof*; 2) verifies the *memb proof*; 3) verifies the *link proof*, that is, all individual proofs are originate from the same credential.
- **Blockchain** stores DID and credential schema based on smart contracts, and maintains an issuance list for each educational institution, providing specified commitment verification path and tree root.

3.2 Architecture

The architecture of the system is depicted in Fig. 3, comprising of several layers: the business layer, API layer, service layer, blockchain service layer, and data layer. Within the blockchain service layer, a smart contract management module has been engineered to facilitate the deployment, updating, and invocation of contracts for DID (Decentralized Identifiers), VCS (Verifiable Credential Schema), Auth (Authentication), and Issuance List. Furthermore, the service layer encapsulates the following functionalities:

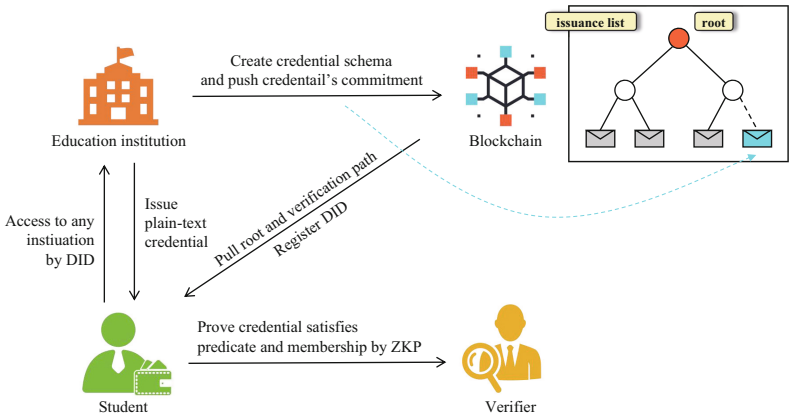

Fig. 2. Overview

- **DID Management.** This involves the creation, updating, and verification of DID identifiers on the blockchain.
- **Verifiable Credential Management.** This encompasses the management of verifiable credential schemas, issuance of verifiable credentials, and their verification.
- **Authority Management.** This includes the granting and revocation of issuing authorities, as well as their verification.
- **zk-SNARK.** This involves defining and compiling predicate circuits (*pred circuits*) and membership circuits (*memb circuits*), setting up Common Reference Strings (CRS), constructing zero-knowledge proofs, and their verification.

3.3 Smart Contract

Smart contracts enable the implementation of complex logic and real-world applications on the blockchain. However, once a smart contract is deployed, it operates independently and repetitively across all blockchain nodes. As such, it is generally considered that only businesses requiring consensus and reusable logic should implement smart contracts on-chain. Moreover, if issues arise or business logic changes after the release of a smart contract, they cannot be resolved simply by modifying and re-releasing the original contract. Consequently, we have implemented a contract layering and update mechanism in this system. As depicted in Fig. 4.(a), we have divided the contract into three tiers: role management, controller, and data. In the event of a contract upgrade, we only need to update the controller contract address in the role management contract, thereby significantly enhancing the system's scalability. The smart contract of this system primarily comprises five components:

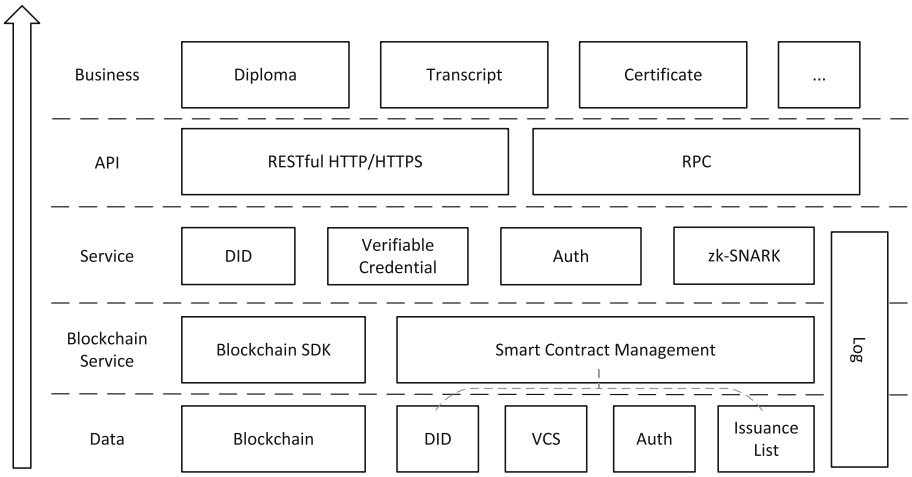

Fig. 3. Architecture

- **DID contract** is responsible for the establishment of the ID structure on the chain, including the generation, reading and updating of DID and DID Document.
- **Verifiable credential schema contract** maintains a key-value pair $<VCSId, VCS>$ mapping table, manages verifiable credential templates, and supports query, addition and update.
- **Role controller contract** is responsible for defining, operating and controlling the authority of DID roles on the chain.
- **Authority contract** is responsible for managing and storing information related to the issuing authority, and supports operations such as checking, adding, deleting, authorizing and revoking the authority institutions.
- **Issuance list contract** maintains a Merkle tree to store commitments with verifiable credentials. Returns the root and verification path for the specified commitments.

The DID contract necessitates the definition of the storage structure and the methods for reading and writing the DID Document. A DID Document is a dataset that describes a DID subject and can be utilized by the DID subject to authenticate itself and demonstrate its association with the DID. The structure of the DID Document is relatively straightforward, comprising a list of disclosed passwords, Authentication, and Service Endpoint. However, with the rapid influx of users, the number of DID will increase significantly. Consequently, designing a large and comprehensive mapping table is impractical due to the immense addressing overhead it would entail. To address this issue, We have implemented the Linked-Event [16] mechanism.

The main idea is to implement the linked list data structure to store and access DID documents, and trigger update events when the documents are modified. In a blockchain network running on Ethereum Virtual Machine (EVM),

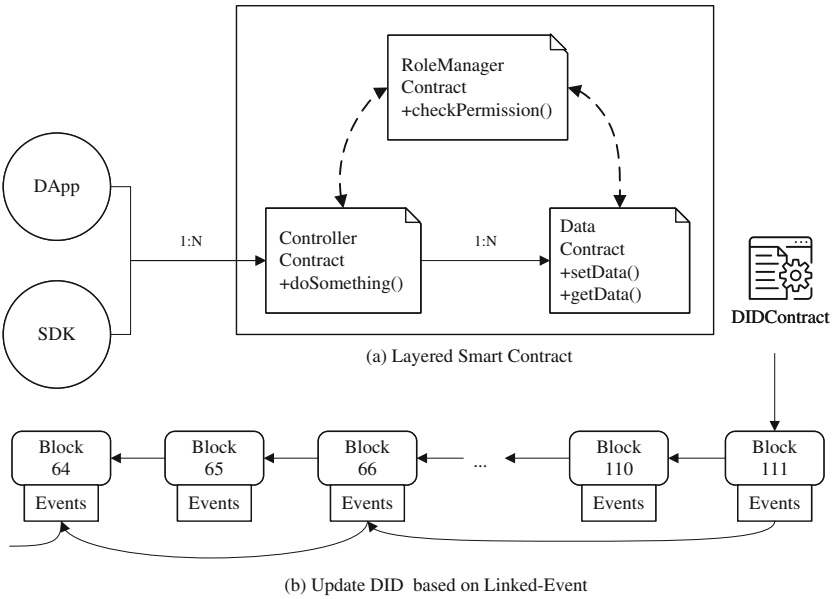

Fig. 4. Smart Contract Framework

every block contains a designated area for storing events related to that block, which are eventually added to the event log. Therefore, when a DID document is updated, an update event can be triggered and stored in the current block's event storage. Additionally, each update event is indexed using the current block height, as depicted in Fig. 4.(b). To retrieve the complete DID document, the corresponding block's events can be traversed in reverse order based on the index.

We have designed the issuance list as a contract that stores all the commitments for credentials. The authorities are required to append leaf nodes to the Merkle tree in order to prove the issuance of the credentials. This enables any external auditor to download the complete list, reconstruct a local copy of the Merkle Tree, and validate the authenticity of the issued credentials. Our design utilizes a simplified Merkle tree structure. The contract status section solely maintains a string array for storing the credential commitments, without preserving the structural information of the tree. During the creation and verification process of the verification path, we calculate the parent-child relationship between nodes by utilizing the node's index in the list and the height of the tree. The specific procedure is outlined in Algorithm 1. The addition of -1 to *path* indicates the left and right positions of the current node in relation to its parent

node. This design effectively reduces the storage overhead of the contract while ensuring efficient construction and verification of the verification path.

Algorithm 1: Get verification path for target node

Input: Merkle tree *leaves*, target leaf *node*
Output: Merkle tree verification *path*

```
path ← null
index ← leaves.indexOf(node)
if index = −1 then
    return path
end
level ← leaves
siblingIndex ← (index % 2 = 0) ? index + 1 : index − 1
while level.size() > 1 do
    siblingIndex ← (siblingIndex = level.size()) ? siblingIndex − 1 :
    siblingIndex
    if siblingIndex % 2 = 0 then
        path.add(level[siblingIndex])
        path.add(-1)
    end
    else
        path.add(-1)
        path.add(level[siblingIndex])
    end
    level ← getNextLevel(level)
    nextLevelIndex ← siblingIndex / 2
    siblingIndex ← (nextLevelIndex % 2 = 0) ? nextLevelIndex + 1 :
    nextLevelIndex − 1
end
return path
```

4 Experiment

4.1 Implementation

We implemented the prototype of the system based on 9.3k lines of Java code and 1.1k lines of Solidity code. The relevant smart contracts are deployed on an open source alliance chain CITA. The code for communication with CITA relies on CITA-Java-SDK. We implemented a common prove and verify interface based on the Groth16 and LinkG16 [8] protocols and encapsulated it as a zk-SNARK module. Groth16 requires a one-time trusted setup to generate a set of parameters called a CRS for each statement (a.k.a., circuit). Once this CRS is generated, it can be used throughout the lifetime of the system to prove different instances of the statement.

4.2 Evaluation

All tests were performed on a server with an Intel Xeon Gold 6330 CPU with physical cores, 42.00 MB cache and 64 GiB memory, running Ubuntu 18.04 with kernel 5.4.0-139-generic. Each figure and table shows the median running time for 100 executions. In addition, the height of the Merkle Tree in our simulated issuance list contract is 32.

We assume a scholarship review scenario. Students need to prove to the review committee that the scores s_i in each subjects on their transcripts are greater than 60 points in order to be eligible for review. Concretely, the access predicate is:

$$\forall i, s_i \geq 60 \wedge expriy \ > \ today, 1 \leq i \leq n \tag{1}$$

where n is the number of subjects, and $expriy$ is the validity period of the credential.

For each system role in Fig. 2, we tested their running time for each operation in this scenario, as shown in Table 1. Among them, the operations of educational institutions pushing credential's commitment and students registering did take significantly more than 3 s. This is because both operations include the time to wait for the receipt of the transaction on the chain, and CITA's block time interval is exactly 3 s. Second. It takes a long time for students to prove credential by generate zk proof, which involves multiple operations in the zk-SNARK module, as shown in Table 2. We now further elaborate on these data and the reasons that influence them.

Before students construct a $link\ proof$, they must first construct $memb\ proof$ to prove that the transcript credential exists in the on-chain issuance list. In this process, an important parameter that affects the proving time and proof size is the depth of the Merkle Tree, as shown in Fig. 5(a).

Secondly, students need to construct $pred\ proof$ to prove that the individual subject scores in the transcript is greater than 60 points. The number of subjects in the transcript and the data type of the scores (such as integer or float) are different, which will affect the proving time and proof size. According to the characteristics of zk-SNARK, the more complex the proving logic is, the more gates exist in the circuit, and the larger the proving time and proof size are. We use the MultiAND circuit as an illustrative example. By progressively augmenting the number of inputs within the circuit, the total number of constraints in the circuit can be expanded. The evaluation results are presented in Fig. 5(b). As the number of circuit constraints increases, there is a corresponding escalation in compile time, Tau time, and prove time. However, owing to the succinct nature of zk-SNARK, both proof size and verification time are consistently maintained at a low level.

We assessed the influence of the quantity of DIDs on the registration and updating time of DIDs. Illustrated in Fig. 6(a), the utilization of Linked-Event technology ensures that, even with a continual rise in the number of DIDs within the system, the registration and update time of DIDs remains consistently around 3.2 s. It is noteworthy that the block interval of CITA is set at 3 s. Additionally, we investigated the impact of the number of DID events on

query time, as depicted in Fig. 6(b). A linear relationship is observed, where an increase in the number of events necessitates the system to query more blocks to acquire comprehensive data. It is essential to highlight that, in practical applications, the frequency of updates to DIDs is generally minimal, thereby having an insignificant impact on the user's query experience.

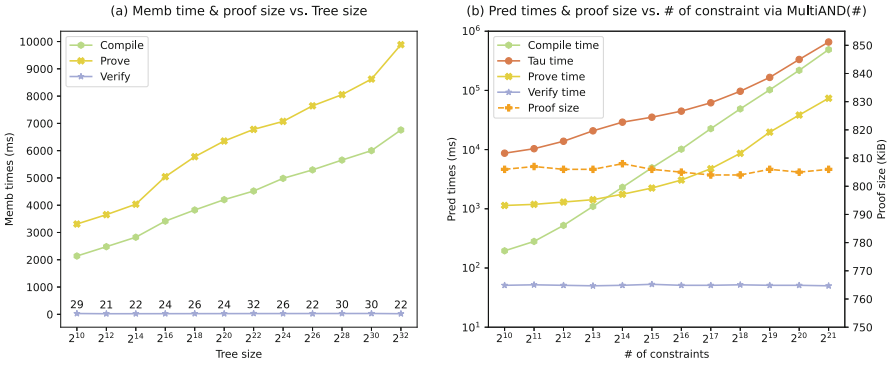

Fig. 5. Circuit Evaluation

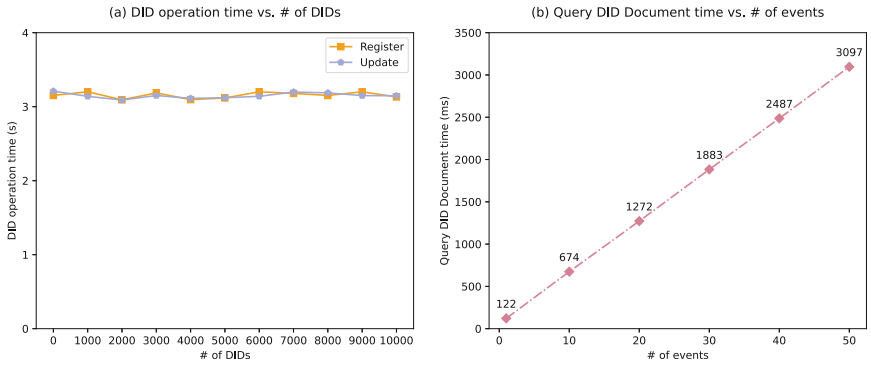

Fig. 6. DID Evaluation

Table 1. Time cost of each operation in scholarship review

Role	Operation	Cost Time (ms)
Education institution	create credential schema	107.02
	issue plaint-text credential	63.21
	push credential's commitment	3095.11
Student	register DID	3186.53
	prove credential by generate *link proof*	11290.35
Verifier	verify *link proof*	73.36
Blockchain	generate root and verification path	154.73

Table 2. zk-SNARK module performance

memb circuit and CRS	*memb proof*	*pred circuit* and CRS	*pred proof*	link proofs
7.35 s	10.13 s	801 ms	1.16 s	162.34 ms

5 Conclusion

This paper introduces a privacy-preserving educational credentials management scheme that leverages decentralized identifier and zero-knowledge proof. The scheme allows educational institutions to issue versatile educational credentials in an efficient manner by standardizing and managing credential templates. These credentials are structured as commitments, tied to specific decentralized identifiers, and stored on a blockchain. Students simply register a DID and can share their educational credentials directly with any supported educational institution or verifier. Building on this, by integrating zero-knowledge proofs with Merkle tree membership proofs, it can be demonstrated that the attributes of the blockchain-stored educational credential satisfy certain predicates. The verifier can cross-check the Merkle tree root and verification path provided by the student against the blockchain to ensure authenticity. We have implemented and deployed a prototype system of this scheme on a consortium blockchain, and have evaluated its performance. Experimental results indicate that the privacy-preserving educational management system based on zero-knowledge proofs has significant practicality and application potential. In future work, we aim to develop a secure credential backup mechanism to maintain data integrity in case of an identity wallet failure. Additionally, we plan to employ more advanced cryptographic primitives to optimize the performance of the zero-knowledge proof module, balancing proof size, computational cost, and verification cost for broader applicability in educational contexts.

Acknowledgment. This work was supported by National Natural Science Foundation of China (No.61972152).

References

1. Qifeng, S., Cheqing, J., Zhao, Z., et al.: Blockchain technology: architecture and progress. J. Comput. Sci. **41**(5), 969–988 (2018)
2. Arenas, R., Fernandez, P.: CredenceLedger: a permissioned blockchain for verifiable academic credentials. In: 2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC), pp. 1-6. IEEE (2018). https://doi.org/10.1109/ICE.2018.8436324
3. Arndt, T., Guercio, A.: Blockchain-based transcripts for mobile higher-education. Int. J. Inf. Educ. Technol. **10**(2), 84–89 (2020)
4. Han, M., Li, Z., He, J., et al.: A novel blockchain-based education records verification solution. In: Proceedings of the 19th Annual SIG Conference on Information Technology Education, pp. 178–183 (2018). https://doi.org/10.1145/3241815.3241870

5. Turkanovic, M., Holbl, M., Kosic, K., et al.: EduCTX: a blockchain-based higher education credit platform. IEEE Access **6**, 5112–5127 (2018). https://doi.org/10.1109/ACCESS.2018.2789929
6. Mishra, R.A., Kalla, A., Braeken, A., et al.: Privacy protected blockchain based architecture and implementation for sharing of students credentials. Inf. Process. Manag. **58**(3), 102512 (2021). https://doi.org/10.1016/j.ipm.2021.102512
7. Du, R., Ma, C., Li, M.: Privacy-preserving searchable encryption scheme based on public and private blockchains. Tsinghua Sci. Technol. **28**(1), 13–26 (2022). https://doi.org/10.26599/TST.2021.9010070
8. Rosenberg, M., White, J., Garman, C., et al.: zk-creds: Flexible anonymous credentials from zk-SNARKs and existing identity infrastructure. In: S&P, pp. 1882–1900 (2023). https://doi.org/10.1109/SP46215.2023.10179430
9. CITAHub: CITA: Cryptape Inter-enterprise Trust Automation (2021). https://docs.citahub.com/zh-CN/0.25.0/cita/cita-intro
10. Sporny, M., Longley, D., Sabadello, M., et al.: Decentralized Identifiers (DIDs) v1.0 (2021). https://w3c.github.io/did-core/
11. Sporny, M., Longley, D., Chadwick, D.: Verifiable credentials data model v1.1 (2022). https://www.w3.org/TR/vc-data-model/
12. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof-systems (extended bstract). In: Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing (STOC), pp. 291–304. ACM (1985). https://doi.org/10.1145/3335741.3335750
13. Bitansky, N., Canetti, R., Chiesa, A., et al.: From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, pp. 326–349 (2012). https://doi.org/10.1145/2090236.2090263
14. Groth, J.: On the size of pairing-based non-interactive arguments. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 305–326. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_11
15. Bunz, B., Bootle, J., Boneh, D., et al.: Bulletproofs: short proofs for confidential transactions and more. In: 2018 IEEE Symposium on Security and Privacy (SP), pp. 315–334. IEEE (2018). https://doi.org/10.1109/SP.2018.00020
16. WeBank: WeIdentity Document (2021). https://weidentity.readthedocs.io/zh-cn/latest/docs/weidentity-contract-design.html