Enhancing Handle System for Digital Credentials Verification by Using Blockchain

Mahamat Ali Hisseine, Deji Chen and Yang Xiao

***Abstract*—The fake document phenomenon has become a major threat in the field of education in today's world of digitalization. Handle System is mostly famous for revolutionizing digital object management and identification. It could be a solution but the concern regarding security and privacy affect the functionality of this system. This paper introduces the Blockchain-based Persistent Identifier for Diplomas (BCPIDD), which enhances the Handle System with Merkle Trees for data integrity, Tendermint for secure consensus, and IPFS for decentralized storage. Comprehensive experiments were conducted to evaluate the system's performance under various conditions, including Distributed Denial of Service (DDoS) attacks and data tampering scenarios. The results demonstrate that the proposed system maintains high throughput and low latency while effectively detecting and mitigating attacks. This analysis confirms BCPIDD's efficiency, consistently outperforming the traditional Handle System (THS) and leading blockchain platforms in ensuring reliable and secure digital identity management.**

***Index Terms*—Blockchain, Handle System, Persistent identifiers (PID), Merkle Trees, Tendermint, IFPS.**

I. Introduction

DIGITALIZATION, as an evolution process, is accelerating and has impacted all aspects of people's lives and work, including educational systems [1]. Various advantages may be gained from digitalization. By digitizing data, costs and time can be considerably reduced. Additionally, businesses can gather information automatically by substituting manual methods with software, so they can better understand process performance, cost drivers, and risk causes. Furthermore, substituting paper and manual procedures with software enables firms to gather data instantly, which can then be analyzed to better understand operation efficiency, expenses, and risk factors [2]. The digital era is transforming industries by leveraging technology to create new opportunities and enhance collaboration [3]. Multiple organization units may utilize data simultaneously when it is digitalized; otherwise, acquiring the data requires a longer process. Digitalization has been efficiently applied as a fundamental approach throughout all aspects of the education system [4]. Digital certification is transforming educational institutions around the world significantly and constitutes a fundamental need for all sectors [5]. It has transformed the way that credentials such as diplomas and certificates are issued, stored, and verified. Digital credentials help connect the recipient to relevant career opportunities through recognized skills, capabilities, and achievements. It offers a quick and reliable way for employers to determine whether an applicant has the right skill sets for the job [6]. In keeping data about the credentials connected with jobs and career achievement, students will be informed in a better way by the decisions concerning educational institutions, and employers can track the endorsement of any new certificates by different organizations [7]. Even in this quickly developing world when everything is moving paperless and being digitalized, at some point in time we observe new challenges that occur, particularly the risk of forgery and the need for secure, tamper-proof systems that can authenticate these credentials with high reliability. Digital credentials include recipient information, which raises questions about data security.

Blockchain is regarded as a promising technology to overcome these issues in the current certification and verification systems. It is a distributed ledger that provides a secure and transparent process to record and transfer data. Furthermore, it generates a decentralized network of computers that all work together to keep a shared ledger [8]. This decentralized nature eliminates the need for intermediaries or middlemen, reducing the risks linked with third-party interventions [9]. Among the many advantages of blockchain, immutability is a particularly useful one, if there's any data stored in the chain, it cannot be altered anymore. Blockchain has demonstrated its ability to enhance data traceability and transparency, ensuring tamper-proof records and reducing risks of unauthorized modifications [10]. It offers a solution to transform the digitalization of credentials by providing a secure, transparent, and decentralized framework to manage and verify. This process guarantees the authenticity of the recorded data. Any credential recorded cannot be altered or tampered with, providing a high level of security and integrity. Several blockchain-based platforms have been developed to check credentials. Each of these systems has its advantages, and limitations [11]–[13]. The adoption of blockchain offers a way of transforming the digitalization of credentials into something people can trust and directly verify from anywhere. It ensures that all the data recorded is genuine, enforcing authenticity. The credentials are stored as a strong hash that cannot be tampered with, ensuring their integrity and security [14], [15]. Despite those considerable advantages, challenges remain, particularly in terms of standardization, scalability, revocation process, and the capacity to handle a complex educational process [16]–[18].

On the other hand, the Handle system is widely recognized as a comprehensive system for allocating, managing and parsing persistent identifiers of digital objects and other resources on the Internet. A persistent identifier (PID) is a long-lasting reference to a digital resource. Compared to URLs, that can break, a persistent identifier can be permanently linked towards the digital entity [19]. The

Manuscript received September 28, 2024; revised March 16, 2025.

Mahamat Ali Hisseine is a PhD candidate of the College of Electronics and Information Engineering, Tongji University, Shanghai 2018040, China (e-mail: mahamat@tongji.edu.cn).

Deji Chen is a Professor of the College of Internet of Things, Wuxi University, Wuxi 214105, China (e-mail: dejichen@tongji.edu.cn).

Yang Xiao is an Assistant professor of the College of Electronics and Information Engineering, Tongji University, Shanghai 2018040, China (e-mail: xiaoyang@tongji.edu.cn).

system includes a developed protocol set, identifier space, and protocol implementation. Additionally, in this system, the user can find and access the information even if the location changes [20], [21]. However, the Handle System is facing a privacy concern due to its capacity to secure the data. It is challenging for the systems to guarantee the data remains safe, and the centralized architecture also poses potential risks such as a single point of failure and vulnerability to cyberattack [22], [23]. This paper presents an approach to address the challenges with the introduction of a Blockchain-based Persistent Identifier for Diplomas (BCPIDD). BCPIDD is built by enhancing the security of Handle System with blockchain. Additionally, the Merkle Trees is deployed for data integrity checks, Tendermint for a faster and safer consensus algorithm, and storing files in IPFS. All these features are put together to form a secure and scalable solution to validate and manage digital degrees that overcome some of the weaknesses intrinsic to the centralized solutions and existing blockchain-based approaches. In other words, BCPIDD tries to fill the deficiency in the existing blockchain-based system, which presents a scalable and trusted framework for managing diplomas. Based on the two described technologies, this paper discusses their potential integration into BCPIDD and what this would mean toward setting up a new standard in managing digital academic credentials at a time when ways will be sought to ensure integrity and trust as we transit into a digital age. The contributions and innovations of this paper are as follows:

1) We introduce the BCPIDD, a blockchain-based Persistent Identifier for Diplomas, an association of blockchain and the Handle System. The model uses blockchain to enhance the Handle System and eliminate the risk of a single point of failure. It makes the integration of the Handle System to provide persistent identifiers to diplomas more secure.

2) A model combines the Merkle Trees for data integrity, Tendermint for secure and efficient consensus, and IPFS for decentralized storage. It provides a comprehensive framework that addresses the limitations such as scalability and revocation of blockchain-based diploma systems.

3) We analyzed and discussed the model's performance on, latency, throughput, response time, network overhead, and resource usage. The performances of BCPIDD are compared with the traditional Handle System and the existing blockchain platforms, such as Hyperledger Fabric, Ethereum, and Quorum. The analysis demonstrates the effectiveness of the proposed model. We demonstrate how BCPIDD can enhance the scalability and security in the management of the digital credentials process and can address the limitations faced by existing systems.

The remaining of this paper is organized as follows. We will give an overview of Handle System and will introduce some related work in Section II. In Section III the design of BCPIDD will be discussed in detail. The experiments and performance evaluation of BCPIDD will be given in Section IV. Finally, Section V will conclude the whole paper.

II. Handle System Basics and Related Work

A. Handle System Basics

The Handle system is a comprehensive distributed system designed for the allocation, management, and resolution of persistent identifiers of digital objects and other resources on the Internet. It is developed by the Corporation for National Research Initiatives (CNRI). It is usually used for managing digital objects in a variety of domains, including academic publishing, digital libraries, Industrial Internet, and other internet-based services. The Handle System ensures that the digital objects can be reliably identified, located, and accessed, regardless of any change in their physical location, ownership, or other attribute [24], [25].

1) Organization of Handle System: Currently, the management is under the control of a Foundation called DONA. It established the Multi-Primary Administrator (MPA) structure to enhance scalability and reliability. The MPAs are globally authorized entities responsible for managing specific regions (See Fig.1. for illustration).

Fig. 1. MPAs repartition across the world

Each country itself also manages a special prefix assigned to it, alike Saudi Arabia (22), China (prefixes 86 and 108), Tunisia (44), Rwanda (25), South Africa (27), USA (20), England (10), Russia (77), and Germany (21). Although each of them is an autonomous architecture, all collaborate to maintain stability, scalability, and reliability in the Handle System globally.

2) Architecture of Handle System: The Handle system has a hierarchical service model. The top level the is GHR (Global Handle Registry), where information is distributed and scalable. The low level consists of several local services, the LHS (Local Handle Service). Fig. 2. given an illustration. The LHS provides identifiers and resolution services in a given region and it is under the control of the GHR [27].

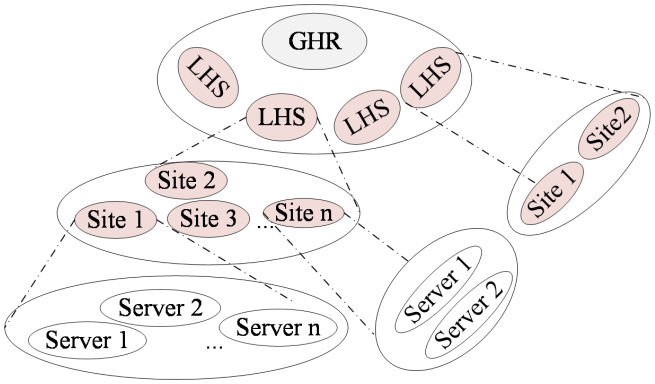

Fig. 2. Handle System service architecture

Handle System is a gathering of handle services, each of which consists of one or more mirror sites (Site 1, Site 2, . . . , Site n), each of which may have one or more servers (Server 1, Server 2, . . . , Server n).

3) Handle Namespace : The identifiers of the Handle System are composed of a prefix and a suffix, separated by a slash (/). The prefix is assigned by the GHR, while the suffix is attributed by a specific LHS. Take an example of a given handle, "86.0001/000202410", "86.0001" is the prefix, and "000202410" is the suffix. This namespace system ensures that identifiers are unique across the whole architecture. And permits a wide range of management of digital objects, from research datasets to digitalized certificates, and provides persistent access to these objects over time [28].

B. Related Work

The security of data has become a matter of utmost concern due to the evolution of the internet. Illegitimate users can easily alter the sensitive information [29]. Blockchain addresses key challenges in decentralized ecosystems, including data security, privacy, and eliminating the single points of failure. Its use to manage and verify academic credentials has gained significant attention in recent years. Several systems have been developed to address the challenges of credential forgery, inefficient verification processes, and centralized vulnerabilities in existing systems. This section reviews some of the key blockchain-based solutions and their limitations.

For instance, BCdiploma is a turnkey blockchain solution for bringing tamper-proof credentials. It allows to easily issue digital badges and certificates to students, employees, and learners, who can then share their accomplishments globally [30]. Blockcerts is an open specification for Blockchain credentials [31],that generate and validate blockchain-based certified documents. It was designed by the Massachusetts Institute of Technology to offer a framework, that allows users to create their decentralized applications which will subsequently check documents [32], [33]. EduCTX is an additional effective approach t for the management of certificates and checking. It offers a worldwide recognized, decentralized educational system that can provide everywhere a connected perspective for students and institutions of higher learning, in addition to different prospective users, such as businesses [34]. Furthermore, several studies have highlighted the benefits of applying blockchain in education, such as enhanced security, transparency, and verifiability of academic credentials [35]–[39].

In the contemporary landscape of employment, managing academic credentials and experiential records is crucial for recent graduates [40]. The digitization and deployment of blockchain in the education area give students the possibility to take part in the evolution of the current technologies essential to their potential careers [41]. Despite the advantages of security and ownership control, existing platforms face challenges such as scalability, the high costs associated with Bitcoin-based transactions, and the difficulty of revoking certificates [16], [17], [42]. To address the limitations of existing systems, this paper introduces a Blockchain-based Persistent Identifier for diplomas.

III. Proposed Model

A. Proposed Model Overview

By combining handle with Blockchain, it is possible to standardize the existing certificate checking platforms and systems that exist today. Even in case the URL of the website is changed or even if the certificate itself is no longer available, the access link to the diploma or other certificate will stay unchanged. Using this method, the identification of the diploma can be accomplished in a much more straightforward manner: the user will obtain the original record by typing the identifier on it. After the document has been distributed, the traceability and life monitoring of the file are made easier to accomplish. Additionally, the handle system will include characteristics and directions for keeping track, supervising, and saving the information as well as issues with persistent identifiers that have been assigned to resources.

Decentralization, traceability, non-repudiation, and non-tampering are all features that the framework offers. It is possible for anyone to verify the academic abilities and certificates that a student has acquired. When candidates are being hired, businesses have the ability to evaluate their qualifications directly, rather than having to pay a specific company to review them. The objective is to mandate the adoption of persistent identifiers at all educational institutions across the globe. The handle system has the potential to bring about a standardized method for certifying certificates all across the world and from any particular location.

In this paper, BCPIDD, a Blockchain-based persistent identifiers that combined the Merkle tree and Tendermint to verify diploma is proposed. The overall architecture of the proposed model is shown in Fig.3. A Merkle tree is a hash-based data structure that is a generalization of the hash list.

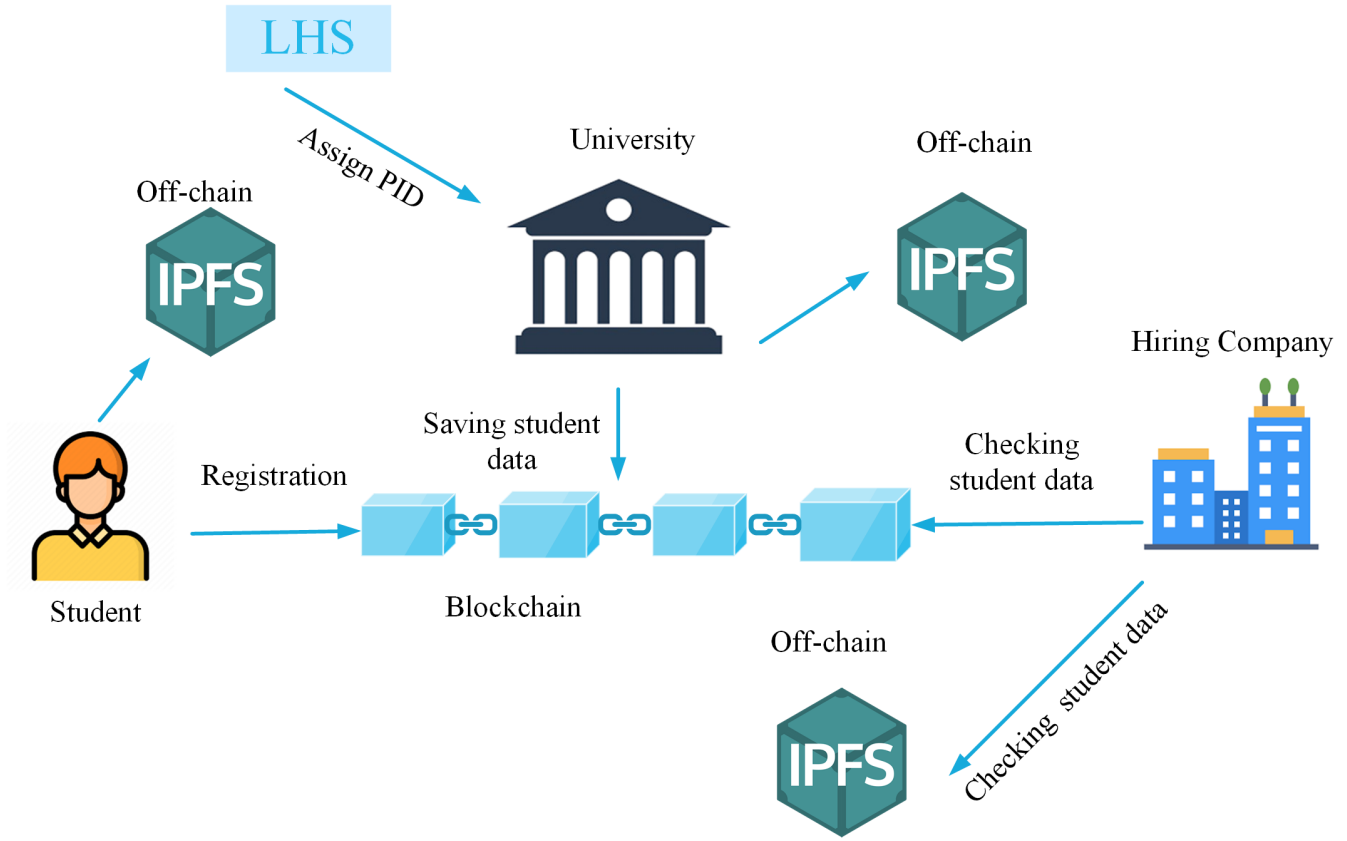

Fig. 3. Architecture of the proposed model

It checks the integrity of large amounts of data. It is used to ensure the integrity of diplomas stored in the decentralized storage. Additionally, it allows fast and secure verification of data in the blocks without requiring access to the whole data. That is crucial for the management of records. Tendermint is a Byzantine Fault Tolerant (BFT) consensus algorithm. It is suitable for decentralized systems, like IPFS that interact with external data storage. It ensures that all nodes in the network agree with the order and the validity of transactions, making it a robust solution for maintaining the integrity of blockchain entries in the Handle System. Additionally, IPFS is associated as an off-chain storage for suitable scalability.

B. Design

(1) Data Preparation and Initialization of Hash Phase

• Diploma Data Structuring

The first step in enhancing the Handle System to address the single point of failure issue involves the careful structuring of diploma data. The diploma data, representing a student's educational achievements, must be organized in a manner that ensures it can be securely stored, verified, and retrieved within the blockchain-based system. The goal is to create a standardized and comprehensive data structure for each diploma, ensuring that all necessary information is included and properly formatted for cryptographic processing. The diploma data D_s is represented as a composite digital document containing the following fields:

$$D_s = \{\text{PID}_s, \text{Student info}, \text{Diploma details}, \text{Date of issue}, \text{Issuer}\} \quad (1)$$

Where, the diploma D_s includes various fields like the Persistent Identifier PID_S, assigned to the student. This identifier is unique and serves as a digital signature representing the student within the system. The student information is information about the student, such as name, student ID, date of birth, and any other relevant personal details such as background and progress, enrollment pattern, strength of staff, and activities undertaken in the schools. The diploma details are the information about the diploma, including the degree awarded, major or specialization, grades, and any honors or distinctions received. The issue date is the official date on which the diploma was delivered. Issuer information is the details about the institution that delivered the diploma, including its name, accreditation, and relevant contact information.

The diploma data is first collected and formatted into this structured format. This structure allows for the consistent handling of diploma data across various institutions and ensures that all necessary information is present for subsequent cryptographic operations.

The accuracy of managing this data in a unified way is crucial for the integrity of Handle System. It allows for safe hashing, storing, and retrieving within the blockchain without access and changes to unauthorized parties. Besides, this Persistent Identifier PID_S uniquely ties the diploma to the student, hence allowing the Handle System to safely resolve and authenticate each of them. This structured framework enhances the security not only of the diploma data itself but also ensures consistency for hashing, storing in the blockchain, and even verification processes. It, therefore, lays a foundation for an effective and robust system that will not only manage educational credentials safely but also preclude any single point of failure from the very base by meticulously

structuring of data.

Our next step is to explain how this organized diploma data is safely hashed with integrity within the blockchain-based Handle System.

• Initial Hash of Diploma Data

Once the data of the diploma D_s are structured, the next most important step is to create an initial cryptographic hash of the data of the diploma. It acts like a unique and fixed-size representation of the whole diploma, encapsulating all details of the diploma in one secure form. The objective here is to get tamper-proof digital fingerprinting of the data of the diploma D_s such that alteration in data can be easily detected. The structured diploma data Ds is passed through a cryptographic hash function $\mathcal{H}$, to generate the initial hash $H_0(D_s)$:

$$H_0(D_s) = \mathcal{H}(D_s) \tag{2}$$

It is a one-way function and takes an input. In our case, the data of the diploma is the input, and compute a fixed-size string of bytes. Even a little modification in the input data would end with a completely different hash, which gives integrity to the data. This initial hash $H_0(D_s)$ is going to be extremely important since it will allow for a secure reference point to the diploma. It is stored to ensure that, later on, the integrity of the diploma data would be checked using a newly created hash with the one stored.

• Include Salt for Added Security

A salt is added before hashing to enhance the security of the hashed diploma data . The term salt simply refers to a random value attached to data before its hashing to avoid precomputed hash attacks, such as rainbow tables. It contains a salt because even if two diplomas do have the same information, their hashes would be distinct, making it more difficult for an attacker to take advantage of the system. To add this, a random value of salt is generated and concatenated with the initial hash $H_0(D_s)$. The hash operation has to be done again on the concatenated data to produce the salted hash $H_s(D_s)$:

$$H_s(D_s) = \mathcal{H}\left(H_0(D_s) \parallel \text{Salt}\right) \tag{3}$$

A random salt value is generated for each diploma. The inclusion of Salt in these hash values will be quite useful for the security of the system; this way, when the underlying data of the diploma itself is identical in more than one instance, the hashes will be different.

• Upload to IPFS

The hashed diploma information $H_s(D_s)$ is stored in the Interplanetary File System. It splits the data into smaller fragments and distributes it through a network of nodes in a decentralized manner. Using IPFS for storing the data of the diploma gives more scalability to the system. This means that the blockchain needs to store only compact, fixed-size content identifiers, which significantly reduces the data load on the blockchain and, hence, enhances the general performance of the blockchain system. In such a decentralized approach, the system is guaranteed to scale efficiently with an increasing number of diplomas while preserving both the integrity and accessibility of the information. A content identifier, C_{IPFS}, is generated by taking the cryptographic hash of the stored information. The above identifier ensures that the data will be uniquely referable and securely accessible from the IPFS network.

$$C_{\text{IPFS}} = \mathcal{F}_{\text{IPFS}}(H_s(D_s)) \tag{4}$$

Where, C_{IPFS} is the content identifier returned by IPFS, uniquely identifying the stored diploma data, $\mathcal{F}_{IPFS}(H_s(D_s))$ represents the function that maps the hashed data $H_s(D_s)$ to its corresponding content identifier in IPFS.

• Leaf Node Creation

This C_{IPFS} is then implanted in the blockchain, which is provided as a leaf node of the Merkle Tree. Each salted hash of each diploma is considered as a leaf node of the Merkle Tree. The Merkle tree is the data structure used among the blockchain systems for verifying the integrity of large datasets in an efficient and safe manner. The goal is to define the base structures the Merkle Tree leaf nodes which will then be used to compute the Merkle Root, representing the entire set of diploma data. And then, each leaf node L_i of the Merkle Tree is assigned a hash:

$$L_i = \mathcal{H}(C_{\text{IPFS}}) \tag{5}$$

Where, $\mathcal{H}(C_{IPFS})$ is the hash function applied to the IPFS content identifier to create the Merkle Tree leaf node. The content identifier C_{IPFS} will ensure that the data linked to the diploma are securely kept in a decentralized storage. The Integration of C_{IPFS} will provide both, IPFS for decentralized storage and the blockchain for immutability. In this context, scalability, data integrity, availability, and resistance to failures are improved.

• Intermediate Hash for Node Pair

The Merkle Tree is built by a recursive hashing of a pair of leaf nodes to their parent nodes. Then, this process is repeated until only one node remains on top, denoted as the Merkle Root. The goal is that a hierarchical hash value structure can be used to efficiently verify any subset of the diplomas for its integrity. Pairs of leaf nodes are combined and hashed to create the first level of parent nodes:

$$P^1_{i,j} = \mathcal{H}(L_i \parallel L_j) \tag{6}$$

Where, $P^1_{i,j}$ is the parent node at the first level, L_i and L_j are leaf nodes, i and j are indices representing some leaf nodes in the Merkle tree. $P^1_{i,j}$ is obtained by hashing the concatenation of two leaf nodes L_i and L_j. And this forms the next level in the Merkle Tree. We create parent nodes from the leaf nodes, which will make a hierarchy to simplify the process of verification. If the data of the diploma changes, then the chain of that tree's parent node changes and finally affects the Merkle Root, thus detecting any tampering.

• Subsequent Parent Node Hashing

The hashing process will continue until only one node remains at the top, known as the Merkle Root. This utility will provide efficient verification. All the levels of this tree are linked with hash values and connected. The goal is to get a complete Merkle Tree structure for a large set of diploma data. This process of combination and hashing for the parent nodes at the intermediate levels is further extended to build the higher level of trees:

$$P^{n+1}_{\kappa,l} = \mathcal{H}(P^m_{i,j} \parallel P^n_{m,n}) \tag{7}$$

Where, $P_{\kappa,l}^{n+1}$ is the parent node at the next level $n+1$; $P_{i,j}^{m}$ and $P_{m,n}^{n}$ are parent nodes at level n; $P_{\kappa,l}^{n+1}$ is obtained by hashing the concatenation of two parent nodes at level n; m, k and l are indices representing the position of the new parent node at level n+1 in the Merkle Tree. This procedure is repeated until the Merkle Root has been reached. The process being recursive guarantees that the Merkle Tree forms a structure whereby any change in the underlying data would affect the Merkle Root. In such a structure, verification of the whole dataset will be efficient and secure; only the Merkle Root will be checked, permitting an efficient and secure verification.

• Final Merkle Root Calculation

The last step in the chain of data preparation and initial hashing is to compute the Merkle Root, which acts as the cryptographic commitment to all the diploma data. It is one hash value that represents the whole tree uniquely. The objective is to produce a single, tamper-evident hash value that encapsulates the integrity of all the diploma data within the system. The final Merkle Root R is computed by hashing the concatenated parent nodes at the highest level of the tree:

$$R = P_{\text{root}} = \mathcal{H}(P_{1,2}^{N} \parallel P_{3,4}^{N} \parallel \dots) \tag{8}$$

Where, R is the Merkle root, representing the cumulative integrity of all diplomas in the system, R is crucial for the Handle System as it provides a secure and efficient way to verify the integrity of all the diplomas stored within the blockchain, P_{root} is the top-most node in the Merkle Tree, formed by hashing together all the nodes at the last level N in the Merkle Tree, N is the last level in the Merkle Tree. The ellipsis "..." typically represents the continuation of this recursive process, here pairs of nodes are hashed together at each level until the final root is obtained. By comparing the Merkle Root at different times or between different nodes, the system can quickly detect any unauthorized changes to the diploma data, thereby resolving the single point of failure issue and enhancing the overall security and reliability of the system.

(2) Blockchain Transaction Preparation Phase

In this step, we transition from securing the diploma data within the Merkle Tree to integrating it into the blockchain. The goal is to prepare, sign, and broadcast the transaction that contains the necessary information to update the blockchain with the latest Merkle Root, thereby ensuring the integrity and immutability of the diploma records.

• Prepare Transaction Data

After calculating the Merkle Root, the next step is to prepare the transaction data that will be recorded on the blockchain. This transaction will serve as a permanent and immutable record of the current state of the diploma data. The objective is to encapsulate the Merkle Root and associated metadata in a transaction that can be verified and added to the blockchain. The transaction data T_x is prepared by combining several key elements:

$$T_x = (T_X\text{ID}, R, \text{TS}, H(B_{n-1})) \tag{9}$$

Where, T_XID is the unique identifier for this transaction, this ID ensures that each transaction can be uniquely referenced within the blockchain. R is the Merkle Root computed in the previous step. This root serves as the cryptographic commitment to all the diploma data, TS is the timestamp represents the exact moment when this transaction is created, the timestamp ensures the chronological ordering of transactions on the blockchain, $H(B_{n-1})$ is the hash of the previous block in the blockchain, this ensures continuity and links the current transaction to the existing blockchain history. The prepared transaction data T_x is critical because it encapsulates all the necessary information to update the blockchain. By including the Merkle Root, the transaction links the diploma data to the blockchain, ensuring its immutability. The timestamp TS and the previous block hash $H(B_{n-1})$ ensure that the transaction is correctly ordered and linked within the blockchain's chain of blocks.

• Sign Transaction

All the transactions must be authenticated by the university before their broadcast into the network. The digital signature is used for his authentication. It will ensure that a transaction T_x is from a verified source, and has not been altered. The goal is to ensure the authenticity and integrity of the transaction by linking it to the source. It is signed by the university's private key $S_{\kappa\mu}$. This digital signature can be expressed by :

$$\sigma_\mu = \text{Sign}(S_{\kappa\mu}, T_x) \tag{10}$$

The digital signature is a cryptographic proof of this transaction. Additionally, it confirms that the process is authorized by the university U. It is crucial for the authentication of any transaction and also the security within the chain. This practice will ensure that only the checked and validated transactions can be stored.

• Broadcast Transaction for Validation

In this step, the transaction is broadcasted into the blockchain to be validated by the other nodes. The transaction will be verified and added to the chain. The goal is to diffuse the transaction T_x across the chain. It is recorded when the consensus is achieved. And then it is broadcasted through all the nodes.

$$B_{cast}(T_x, \sigma_U) \tag{11}$$

The broadcast, B_{cast} is the process of diffusing a transaction through all the nodes within the chain. After the reception of a transaction T_x, each node will check the digital signature σ_u. Then, if it passes the check, the transaction is considered valid, otherwise it is rejected. This decentralized process of validation can prevent the system from a single point of failure since there are several checks by different nodes before adding this transaction.

(3) Consensus Process Using Tendermint Phase

Tendermint is a consensus mechanism that allows the system to launch across the nodes securely and consistently. It works with a Byzantine Fault Tolerance of up to 33%. That means the system based on Tendermint will work even if $1/3$ of the node fails.

• Propose Block with Transaction

After the broadcasting process, the transaction T_x will be included in a new block. This process is a crucial part of the consensus as the participants will decide the validation of this block. The goal is to add the new block that contains the last transaction and diffuse it into the network for validation by the participant through a consensus. The new block B_n can be expressed as,

$$B_n = (T_x, \sigma_U, \text{TS}, H(B_{n-1})) \tag{12}$$

The block B_n will include the transaction T_x, the digital signature σ_U, the timestamp TS, and the hash of the previous block $H(B_{n-1})$. The first process is the proposition of block is the first step toward consensus. It has all the information that the validator needs to know to make a decision on whether to add it to the blockchain or not. This proposal will provide a continuity into the chain.

• Validator Pre-vote Stage

The participants who act as validators will validate the proposed block through a pre-vote process. Before the validation, the block will be checked. Each participant i will perform a pre-vote V_i expressed as,

$$V_i = Pre_Vote(B_n, \sigma_U) \tag{13}$$

The goal for each participant i is to verify the transaction T_x, by checking the validity of the digital signature σ_U, and the block B_n. The pre-vote of participants will indicate if the block is accepted or not. The process will ensure the integrity and authenticity of the chain.

• Validator Pre-commit Stage

The block needs to receive a majority of votes from the participants to pass this phase. After the pre-vote the participant needs to pre-commit, this process will add more security against any attack. And then, if the agreement is achieved, it will enter the phase of pre-commit, which is expressed as,

$$C_i = Pre_{Commit}(V_i) \tag{14}$$

Where C_i, is the pre-commit message sent by a participant i, this represents a formal commitment for the block B_n. After a block receives the majority of Pre-commit, it will be processed.

• Finalizing Consensus

The final decision to include the block into the chain is based on the total number of pre-commits achieved. It will be included after getting the enough number. The decision is expressed as :

$$B_n = \begin{cases} \text{include the block} & \text{if } \sum_{i=1}^{m} C_i \geq \tau \\ \text{rejected} & \text{otherwise} \end{cases} \tag{15}$$

Where, $\sum_{i=1}^{m} C_i$ represents the total number of pre-commits achieved from the participants, m is the total number of participants, i is the index for each participant, τ is the consensus threshold and represents the minimum number required to include the block.

• Adding the Block (B_n)

After the consensus is achieved, in this stage the block B_n, is appended to the chain. The state of the blockchain is updated,

$$\text{Blockchain} \leftarrow B_n \tag{16}$$

Now the chain includes the Merkle root R with the new block B_n. This addition of block is finalizing the process. It provides the Handle System with a secure and decentralized way to store data and resolve its single point of failure issue. The data are distributed across all the nodes of this network, where they are stored following the consensus mechanism.

(4) Verification Process Phase

It involves verification of whether the blockchain-based Handle System will allow verification of the authenticity of the diploma, also if the data on the blockchain concerning the diploma is indeed genuine and has not been tampered with. This step of verification provides prospects for trustworthiness and reliability in the overall system.

• Merkle Root Retrieval

We will start by retrieving the Merkle Root from the blockchain. It is used as the cryptographic summary of all the credentials data stored within the system. We need to find the specific time when this data is added. This Merkle Root $R_{retrieve}$ is expressed as,

$$R_{\text{retrieve}} = \text{GetMerkleRoot(TxID)} \tag{17}$$

Where $TxID$ is the identifier for the transaction that has been stored at a specific time. This Merkle Root $R_{retrieve}$, was included during the storage of the diploma. It is a crucial factor to check this diploma, employed as the reference point to verify the integrity of the data. If there are any differences between the retrieved Merkle Root and the origin that will indicate an alteration was happened with the diploma data.

• Recompute Diploma Hash

The verification of the diploma is done by the system, which needs to rehash the data of the diploma and then compare the hash with the one contained in the Merkle Tree. In such a case, assurance of data integrity is needed, meaning that the data was recorded for the first time without tampering. What is required here is to get a hash from the current diploma data and then compare it with the one stored in the Merkle Tree. The system recomputes the hash of diploma $H_{recompute}(D_s)$ by using the original data of the diploma D_s :

$$H_{\text{recompute}}(D_s) = \mathcal{H}(D_s) \tag{18}$$

Where $\mathcal{H}$ is the cryptographic hash function used to produce the hash. The recomputation of the hash regarding the diploma ensures that the involved diploma data is the same as that which was originally stored on the blockchain. Any difference between this recomputed hash and the hash stored in the Merkle Tree will cast suspicion regarding the veracity of this diploma data.

• Generate Proof Path for Diploma

The entire proof path that a Merkle Tree creates makes it extremely efficient to verify whether certain data exists or not, in this case, the existence of the diploma without needing to go through all the tree. The proof path should illustrate how the hash of the diploma flows into the Merkle Root. We aim to provide a path that proves the hash of the diploma connects with the Merkle Root, hence proving its inclusion in the dataset. It then constructs the proof path Υ_{Ds} starting with the leaf node L_i for the hash of the diploma up through the intermediate nodes to the Merkle Root:

$$\Upsilon_{Ds} = \{L_i, P^1_{i,j}, P^2_{k,l}, \ldots, R\} \tag{19}$$

Υ_{Ds} is the hash in a sequence that forms the proof path from the leaf node of the diploma to the Merkle Root R; L_i is the leaf node, which is the hash of the diploma; $P^1_{i,j}$ and $P^2_{k,l}$ are some intermediate parent nodes that form part of the path; i and j are indices pointing to particular leaf nodes in the Merkle Tree, each carrying part of the hashed data of the diploma; k and l are the indices of parent nodes higher in the Merkle Tree. These parent nodes are created by hashing the concatenation of their child nodes. The proof path Υ_{Ds} is crucial to confirm that a given diploma is indeed

part of the dataset represented by the Merkle Root. It hence gives the possibility to verify the certainty of the diploma's authenticity without actual access to the whole Merkle Tree or the whole dataset.

• Verify Diploma Integrity

The verification of the recomputed hash including its proof path has to correctly result in the Merkle Root. In such a way, it is ensured that the data of the diploma is not tampered with. In general, it shall be proved that the data about the diploma was not manipulated and was correctly included in the Merkle Tree represented by the retrieved Merkle Root. We will check if the Merkle Root calculated from the proof path is equal to the retrieved Merkle Root, to confirm the authenticity of the diploma data,

$$(D_s) = \begin{cases} \text{valid} & \text{if } R_{\text{retrieve}} = \mathcal{H}(\Upsilon(D_s)) \\ \text{altered} & \text{otherwise} \end{cases} \tag{20}$$

This step of verification will ensure that no alteration of diploma data occurs. The evidence of this authenticity and unaltered status of the diploma is achieved if the recomputed Merkle Root matches the retrieved Merkle Root. This last phase gives reliability and trust in the Blockchain-based Handle System to securely manage and check the credentials.

(5) Long-term Preservation Phase

The long-term preservation of diploma data is necessary so that the integrity and authenticity of the same are preserved seamlessly over time. The step focuses on the continuing verification and updating of Merkle Tree and blockchain so that the integrity of data is preserved, and this makes the system robust enough from data degradation or potential tampering.

• Periodic Recalculation of Diploma Hashes

It should be reiterated that, over time, diploma data periodically needs to be re-verified for its integrity. This shall include the recalculation of the hash of each diploma with regard to consistency with the originally stored hash. For every diploma D_s, its hash is periodically recalculated as:

$$H_{\text{check}}(D_s) = \mathcal{H}(D_s) \tag{21}$$

$H_{check}(D_s)$ is the rehashed value of the hash of the diploma data. This process can then be scheduled periodically, with no change in the content of the diplomas over some time. This process will identify any alteration or corruption of data that may have happened since the data was uploaded.

• Recompute Updated Merkle Root

After recomputing and verifying all the diploma hashes, a Merkle Root should again be computed in order to represent the data at its present state. In fact, this Merkle Root represents the integrity of the dataset in accumulation. Here, the goal will be to recompute an updated Merkle Root, showing the present state of all diploma data after periodic verification. The new Merkle Root R_n will be generated from the hash of recomputed diploma hash values:

$$R_n = \mathcal{H}(H_{\text{c}}(D_1) \parallel H_{\text{c}}(D_2) \parallel \cdots \parallel H_{\text{c}}(D_n)) \tag{22}$$

Where R_n is the renewed Merkle Root, and it is the new valid state of data in the diploma. $H_c(D_1), H_c(D_2), \ldots, H_c(D_n)$ are the hash values of all the diplomas in the system computed again. The Merkle Root recalculates to make the integrity of the entire dataset intact. And this new Merkle Root will be utilized on the blockchain to keep an updated record of the data of the diploma in a secure manner.

• Propose New Block with Updated Merkle Root

The new Merkle Root R_n will be stored into the chain. Therefore, a new block B_{n+1} will be created. It will ensure that the chain contains the most current state of the data. This new block is expressed as:

$$B_{n+1} = (R_n, \text{TS}_{n+1}, H(B_n)) \tag{23}$$

Where B_{n+1} is containing the updated Merkle Root R_n; TS_{n+1} is the timestamp of the new block; $H(B_n)$ is the hash of the previous block. The process will ensure that any verification will consider the updated and valid data.

• Validator Pre-vote for Updated Block

Once the new block is proposed, it has to be pre-voted for inclusion into the blockchain by all validators. The network is designed in such a way that every prevote must go through the Tendermint consensus process analysis in order for the block to meet requirements set by the network. It is essentially a set of processes designed for the purposes of collecting pre-votes from validators in an effort to show preliminary approval for the proposed block. The validators cast their prevotes for the new block $B_(n+1,)$,

$$V_{i+1} = \text{PreVote}(B_{n+1}, \sigma_U) \tag{24}$$

Where V_{i+1} is the prevote from the validator at i+1 for the proposed block B_{n+1}, σ_U is the signature of the block proposer verifying the block's authenticity. This stage allows one to make sure that the proposed block is valid and follows all the consensus criteria for adding blocks into a blockchain. By this way, the addition of malicious or invalid blocks into a blockchain can be prevented.

• Validator Pre-commit and Finalize

After successful pre-votes, validators enter the pre-commit phase, whereby they formally commit to the proposed block. This act seals the agreement of the validators to make sure the block is added to the blockchain once it meets the consensus threshold in its finality. What happens here is an attempt to make sure that the commitment of the validators to the block is sealed once it meets the minimum requirements for consensus. Those who pre-voted for the block previously pre-commit:

$$C_{i+1} = Pre_{Commit}(V_{i+1}) \tag{25}$$

Where C_{i+1} is the pre-commit message of validator $i+1$ about the proposed block B_{n+1}. The pre-commit stage now ensures that only those blocks that have broad support among validators are added to the blockchain. This step adds a layer of security since, during the second round, validators have to confirm their previous choices.

• Add Updated Block

The block with the new Merkle Root is added to the chain after getting enough pre-commit from the validators. The block B_{n+1} is then appended to the chain,

$$B_{n+1} = \begin{cases} \text{added} & if \quad \sum_{i=1}^{m} C_{i+1} \geq \tau \\ \text{rejected} & \text{otherwise} \end{cases} \tag{26}$$

Where m represents the total amount of validators in the blockchain network. Each of them is allowed to vote for the inclusion of a new block; i is the index that represents

each validator out of the total m validators; B_{n+1} would be added to the blockchain in case the sum of these votes meets or exceeds a predefined consensus threshold τ. This new block will update the state of the blockchain in such a way that the most updated Merkle Root now forms part of the immutable record. With the addition of this updated block to the blockchain, the data in the Handle System is secured and tamper-proof. This step completes the long-term preservation and ensures that all diploma data is constantly and reliably stored on the blockchain, addressing the problem of a single point of failure.

(6) Diploma Update and Revocation Process Phase

The updating and revoking of diplomas make the System dynamic, adaptive for corrections and updates, and even revocation of a diploma when necessary. The operations are done on the data saved into IPFS.

• Diploma Update Request

A request for updating of the diploma may occur in the case of a change in the information linked to that particular diploma, such as error correction, change in name of the student, change in degree title/diploma information. The objective is to trigger an update process on the blockchain-based Handle System for records of the diploma in question. An Update request U_s is sent from the university,

$$U_s = \text{Update}(\text{PID}_s, D'_s) \tag{27}$$

Where U_s is the update request; PID_s is the unique identifier of the student whose diploma is being updated; D'_s is the new diploma data to replace the existing information. An update request is the initial step to ensure that changes to the educational record are reflected in the Handle System. By allowing updates, the system can help maintain data integrity over time.

• Generate Updated Diploma Hash

After sending the update request, a new hash of the updated diploma data is generated. This should ensure that the updated information is introduced in a secure manner within the framework of the system. It thus, therefore, attempts to compute a new cryptographic hash representative of the updated diploma data. The new hash $H(D'_s)$ from the updated diploma data D'_s applies the cryptographic hash function H involved in the production of the hash,

$$H(D'_s) = \mathcal{H}(D'_s) \tag{28}$$

Because this step involves the hash of updated diploma data, it should be unique, and the new hash will form part of the Merkle Tree to ensure that updated data are kept securely and their authenticity is verifiable on this chain.

• Include Updated Hash in Merkle Tree

The new hash of this updated diploma should be included in the Merkle Tree. This updated data should form part of the one overall cryptographic structure representing all the diploma data. Any updated diploma hash should be aimed to form part of the existing Merkle Tree structure. A newly updated hash $H(D'_s)$ is positioned as a leaf in the Merkle Tree L'_s,

$$L'_s = H(D'_s) \tag{29}$$

Adding the updated hash to the Merkle Tree makes the new data of the diploma verified in a secure yet efficient manner within the blockchain. This would integrate the updated information into the already existing cryptographic structure of the system.

• Recompute Merkle Tree After the Update

Since the refreshed hash is a part of the Merkle Tree, the tree will need to be recomputed to reflect those changes. The whole tree structure should mirror the current state of every piece of diploma data. One should not forget to remake the Merkle Root after adding the renewed diploma data into the tree. The updated Merkle Root R_{update} is expressed as,

$$R_{\text{update}} = \mathcal{H}(L_1 \parallel L_2 \parallel \cdots \parallel L'_s) \tag{30}$$

Where L_1, L_2, and L'_s are the leaf nodes in the Merkle Tree; They will include the updated diploma hash.

• Propose Block with Updated Root

The revised Merkle Root has to be recorded on the blockchain. A block is proposed that contains the updated Merkle Root together with other information. In this paper. A new block is proposed which updates the blockchain with the most recent Merkle Root such that it remains updated. We propose a new block B_{update},

$$B_{\text{update}} = (R_{\text{update}}, \text{TS}_{\text{update}}, H(B_n)) \tag{31}$$

The new block contains the updated Merkle Root R_{update}, the timestamp TS_{update}, and the hash of the previous block $H(B_n)$. The proposal for a new block ensures that the blockchain is updated concerning the current state of the Handle System. This integrates the updated diploma data in the blockchain that forms part of the immutable record.

• Validator Pre-vote for Update Block

Validators should verify the proposed block and pre-vote on whether the inclusion of the block into the blockchain is accepted. This is part of the consensus procedure to decide that the update really happened. The goal here is to get the pre-votes from the set of validators, which indicates preliminary approval to go forward with the updated block. The validators pre-vote V_{update} for the proposed block is expressed as,

$$V_{update} = Pre_{Vote}(B_{\text{update}}, \sigma_U) \tag{32}$$

This pre-vote stage prevents the addition of an invalid or unauthorized update in the system by ensuring that the update is independently checked by validators for integrity within the Handle System.

• Final Pre-commit and Block Addition

The validator commits formally to adding the block to the blockchain provided it passes the final threshold of consensus. The aim at this stage is to summarize the commitment of the validator with the updated block so that it meets the consensus,

$$B_{update} = \begin{cases} \text{added} & if \quad \sum_{i=1}^{m} C_{update} \geq \tau \\ \text{rejected} & \text{otherwise} \end{cases} \tag{33}$$

Where, C_{update} is pre-commit message from the validators concerning the block being proposed, m stands for the total amount of validators in blockchain space. Each validator votes for or against the acceptance of a block addition; i is the index representing each individual validator among the total amount of m validators. This step ensures that the block acquires the majority of validators before its addition to the

blockchain. This is an important pre-commit phase that helps in ensuring the integrity of the system.

• Revoke Diploma Process

In cases where a diploma should be revoked, the system needs to invalidate the hash of the diploma and remove it from the Merkle Tree so it will no longer be verified. It can be used for removing a diploma from the Handle System so that it is no longer valid. Any Merkle Proof linked with the diploma will be marked as invalid in the system. The event of revocation for a given diploma adds another transaction to the blockchain. This will ensure that at any moment in time when it is attempted to verify the said diploma, it will be invalidated. It allows the integrity and immutability of blockchain while enabling the practical function of diploma revocation. To revoke a diploma is to invalidate its Merkle Proof and remove its associated leaf node:

$$\text{Revoke}(\text{PID}_s) = \mathcal{R}(\text{Merkle Proof}) \tag{34}$$

Where $\mathcal{R}(\text{Merkle Proof})$ is the revocation process, which declares the Merkle Path of a diploma invalid and removes the latter from the Merkle Tree. Algorithm 1 is the pseudocode illustrating this revocation process.

Algorithm 1 Revocation of Diploma

Input: Persistent Identifier PID
Output: Revocation confirmation message
Find $H_{D_s} \leftarrow FIND\_Diploma\_Hash(PID_s)$ ▷ Retrieve diploma hash
if H_{Ds} exists in Merkle Tree **then**
  Remove H_{Ds} from Merkle Tree
  Update Merkle Root R_{update}
  Record revocation in Blockchain
  return "Diploma Revoked Successfully"
else
  return "Diploma Not Found"
end if

It constitutes one of the key features for ensuring the veracity and integrity of the Handle System. Every merit certificate that gets revoked has to establish that only valid ones are part of the system; hence, the credentials of such a revoked one get scratched off from the framework.

IV. Experimental Results and Discussions

A. Experiments Setup

The experiment was conducted on Amazon Web Services to host several virtual machines acting as blockchain nodes. These virtual machines were hosted in different AWS data centers geographically dispersed at different locations, thus simulating a geographically distributed blockchain network. This was done in order to model the decentralized nature of a blockchain-based system, in such a way that it would allow the network to act most similarly to real-world conditions in a global academic setting. A Virtual Private Cloud (VPC) was developed to build a secure interaction within AWS. It offers node interaction protection through a private, encrypted network that allows them to connect as they would on an entirely decentralized blockchain. The Tendermint consensus algorithm is adopted in the blockchain design for its efficiency, fault-tolerant, and providing a fast transaction process. Another important integration is the Merkle Trees, which guarantees the integrity of the diploma data through its hashed structure for a simple verification process. The use of IPFS was enabled for decentralized storage; hence, the data could not be lost if some of the nodes went down. Thus, we used the Handle Server, which assigns persistent identifiers to digital diplomas, making access and identification permanent. The Handle server operated as a central component for PID management, synchronizing with the Global Handle Registry (GHR) to ensure global uniqueness and reliability. Communication between the application and the Handle server was facilitated using its Restful API, supported by robust authentication mechanisms involving handle identities and cryptographic credentials. The tests were done at different rates of records. It ranges from small-scale loads of 10 to large-scale datasets of 100,000 records. The key metrics that are measured in this experiment include response time, resilience, consensus time, throughput, and storage efficiency. The real-time performance was tracked with the monitoring tools, Prometheus and Grafana. The first collects real-time performance metrics from the AWS-hosted nodes. The second is used to visualize those metrics in real time. Python was also used in the data analysis and visualization. To assess security resilience, additional configurations were implemented to simulate attack scenarios within this AWS-based environment. For Distributed Denial of Service (DDoS) tests, traffic generation tools like Apache JMeter and hping3 were utilized to overwhelm specific nodes or the entire network with excessive requests. The impact of these attacks on transaction throughput, latency, and node availability was monitored in real-time using AWS CloudWatch and custom logging tools. For data tampering simulations, BCASim, an open-source blockchain simulator for attack analysis was employed. It can simulate unauthorized modifications to transaction data, testing the effectiveness of each system's consensus mechanism. This will help to detect and reject malicious operations.

B. Performance Metrics Analysis

In this section, we perform a systematic performance evaluation of the BCPIDD framework compared to the traditional handle system (THS) and also to several blockchain-based systems including Hyperledger Fabric, Ethereum, and Quorum. We chose these systems as representative solutions across different blockchain architectures, shedding light on permissioned blockchain relevant to our use case. The performance metrics include latency, throughput, response time, time to finality (TTF) storage efficiency, resource utilization (CPU and memory), and network overhead, each of which highlights the strengths and limitations of the various systems.

1) Latency: The latency is the time it takes for a transaction to be processed from initiation to completion. Lower latency indicates a faster system response, which is critical in time-sensitive applications. Fig. 4. shows an illustration of latency for the different systems. With a low latency across all the record sizes, the BCPIDD system shows a clear performance advantage over the traditional Handle System and also the other blockchain-based models. It maintains a value under 0.09 seconds even with 10,000 records. This

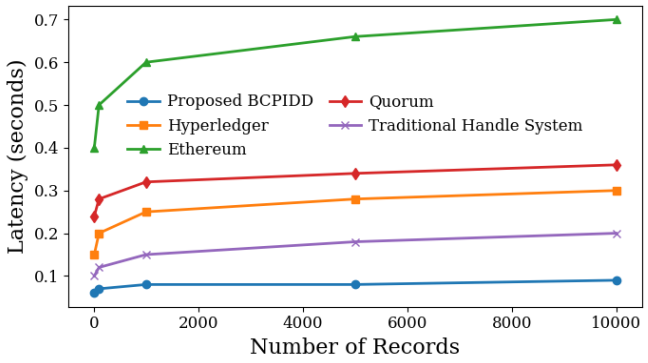

Fig. 4. Latency Comparison

performance can be attributed to the use of the Tendermint consensus algorithm, which provides fast finality and avoids delays typical of traditional consensus methods. In contrast, the traditional Handle System exhibited moderate latency, starting at 0.10 seconds for 10 records and increasing to 0.20 seconds for 10,000 records. The centralized nature of the Handle System introduces delays as requests must traverse hierarchical layers. The model based on Ethereum showed the highest latency, reaching 0.70 seconds for 10,000 records. This can be explained by the batching of transactions and block confirmation processes, which prioritize fault tolerance over speed.

2) Throughput: The transaction throughput is commonly calculated by the number of transactions that can be processed per second (TPS). The throughput comparison across different models is illustrated in Fig.5. BCPIDD levers over 21,600 TPS for 10,000 records. This performance stems from the system's decentralized storage via IPFS and efficient Tendermint consensus, which reduces the computational overhead associated with transaction processing. The traditional Handle System, by contrast, was limited to 4,600 TPS at 10,000 records, reflecting its reliance on a centralized resolution server that becomes a bottleneck under high loads. Ethereum exhibited the lowest throughput, at just 800 TPS for 10,000 records, due to its block production times and consensus mechanism.

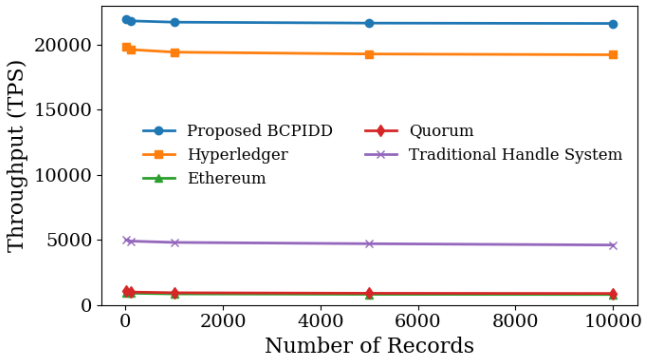

Fig. 5. Comparison of Throughput

3) Response Time: The response time refers to the time takes by the system to respond to a request. It is measured from the instant that the request is sent until the response is received. The response times across different systems are shown in Fig.6. Our proposed system achieves a low response time due to the integration of an efficient consensus algorithm based on Tendermint, that processes transactions with minimal delay. It keeps a response time of

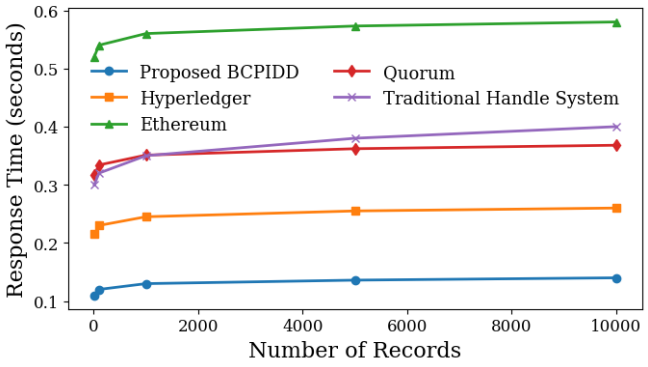

Fig. 6. Response Time Comparison

0.14 seconds for 10,000 records. This performance highlights the effectiveness of Merkle Trees and IPFS in enabling quick and reliable access to credential data. The traditional Handle System, with its reliance on hierarchical resolution, displayed significantly slower response times, reaching 0.40 seconds for 10,000 records. Ethereum was the slowest, with response times of 0.58 seconds for 10,000 records, reflecting the inherent delays in its transaction batching and block finalization processes.

4) Time to Finality: represents the duration required for a transaction to achieve finality within a blockchain network. Finality, in this context, refers to the point at which a transaction becomes immutable in the blockchain's ledger, making it irreversible. An illustration is given in Fig.7

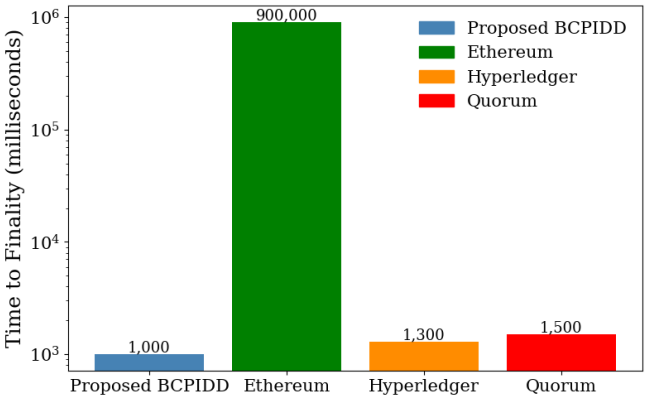

Fig. 7. Time to Finality

With Tendermint consensus, the time to finality is considered instant; meaning a transaction is considered finalized as soon as it is included in a block, achieving immediate finality with no need to wait for further confirmations. Unlike some other consensus mechanisms that offer probabilistic finality, Tendermint provides absolute finality, where a block is considered definitively confirmed once added to the blockchain. The results show that BCPIDD achieves near-instant finality at 1000 milliseconds, allowing immediate transaction confirmation. In contrast, Ethereum takes 9000 milliseconds, while Ethereum (PoS) requires 768 seconds, due to their reliance on block confirmations. Hyperledger

Fabric and Quorum (1 second) also provide quick finality. The findings suggest that deterministic finality, as seen in BCPIDD, ensures faster and more efficient credential verification compared to probabilistic models.

5) Storage Efficiency: Storage efficiency is the measure of how effectively a storage system utilizes available space by optimizing storage resources. A higher percentage indicates a better use of storage. The storage efficiency result is illustrated in Fig.8.

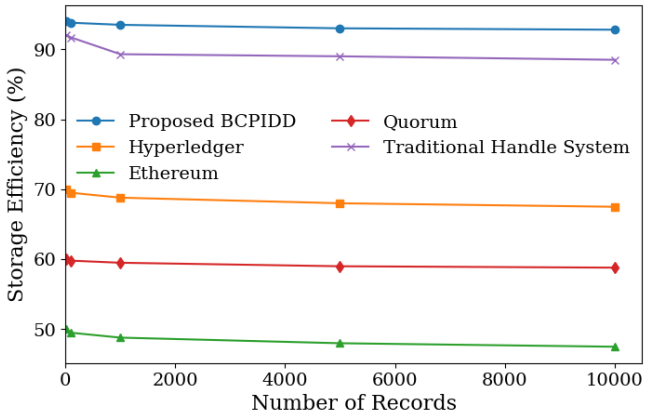

Fig. 8. Storage Efficiency Comparison

BCPIDD demonstrated the highest storage efficiency, maintaining approximately 95% efficiency across all record sizes. This result highlights the benefits of integrating IPFS, which employs content-addressable storage to minimize redundancy and optimize data distribution. In contrast, the Handle System exhibited also significantly high storage efficiency, averaging around 93%. This can be attributed to its centralized storage model. Centralized storage is generally considered more efficient in managing large data volumes and maintaining data consistency due to its single point of control. Ethereum demonstrated the lowest storage efficiency at approximately 50%, primarily due to the high overhead associated with blockchain-based storage mechanisms. Hyperledger Fabric and Quorum performed moderately well, with storage efficiencies of approximately 70% and 60%, respectively. These results underscore the impact of architectural design on storage optimization, with decentralized systems like BCPIDD leveraging advanced techniques to achieve superior performance.

6) Network Overhead: Network overhead provides critical insight into the data transfer requirements of each system. The comparison of consensus time between the different models is illustrated by Fig.9.

BCPIDD exhibited the lowest network overhead, with 120 MB at 10,000 records, due to its efficient use of IPFS for decentralized storage and Tendermint's lightweight consensus. The Handle System, with a slightly higher network overhead of 190 MB, benefits from its centralized architecture but lacks the redundancy and fault tolerance of blockchain-based systems. Ethereum demonstrated the highest network overhead, reaching 260 MB at 10,000 records, reflecting the heavy data replication inherent in blockchain storage and consensus. Hyperledger Fabric and Quorum exhibited intermediate network overheads of 230 MB and 250 MB, respectively, balancing scalability with fault tolerance.

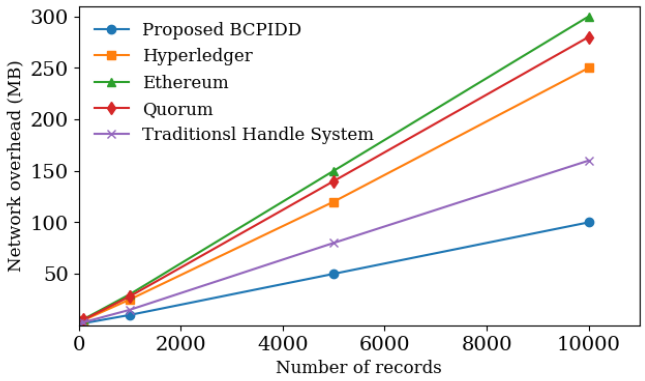

Fig. 9. Network Overhead

7) Resources Usage: The resource usage, encompassing CPU and memory usage, provides a deeper understanding of the computational and storage demands of each system.

The analysis result for CPU utilization is presented in Table I. BCPIDD required only 35% usage at 10,000 records, compared to Hyperledger Fabric at 50%, Ethereum at 70%, and Quorum at 55%. This efficiency stems from Tendermint's streamlined consensus and the offloading of heavy data storage to IPFS. The Handle System, at 40% CPU usage, performed better than most blockchain systems due to its simpler centralized model but lacked the scalability benefits seen in BCPIDD.

TABLE I
CPU USAGE (%) UNDER DIVERSE RECORD NUMBER

Records number	Models				
	BCPIDD	Hyperledger	Ethereum	Quorum	Handle
10	5	10	15	12	8
100	8	15	25	18	12
1000	15	25	35	28	20
5000	25	35	50	40	30
10000	35	50	70	55	40

For memory usage, BCPIDD maintained a gradual increase, reaching 500 MB at 10,000 records. The results of analysis are in Table II. Hyperledger Fabric and Ethereum have a higher memory demands, reaching 750 MB and 900 MB, respectively, due to their complex transaction management and consensus mechanisms. The Handle System, while simpler, required 600 MB at 10,000 records due to its reliance on centralized data storage.

TABLE II
MEMORY USAGE (MB)

Records number	Models				
	BCPIDD	Hyperledger	Ethereum	Quorum	Handle
10	50	100	120	110	60
100	100	200	250	220	130
1000	200	350	400	370	250
5000	350	500	600	550	400
10000	500	750	900	800	600

Fig.10 gives a visual illustration of CPU and memory usage. It shows that BCPIDD maintained the most efficient resource usage.

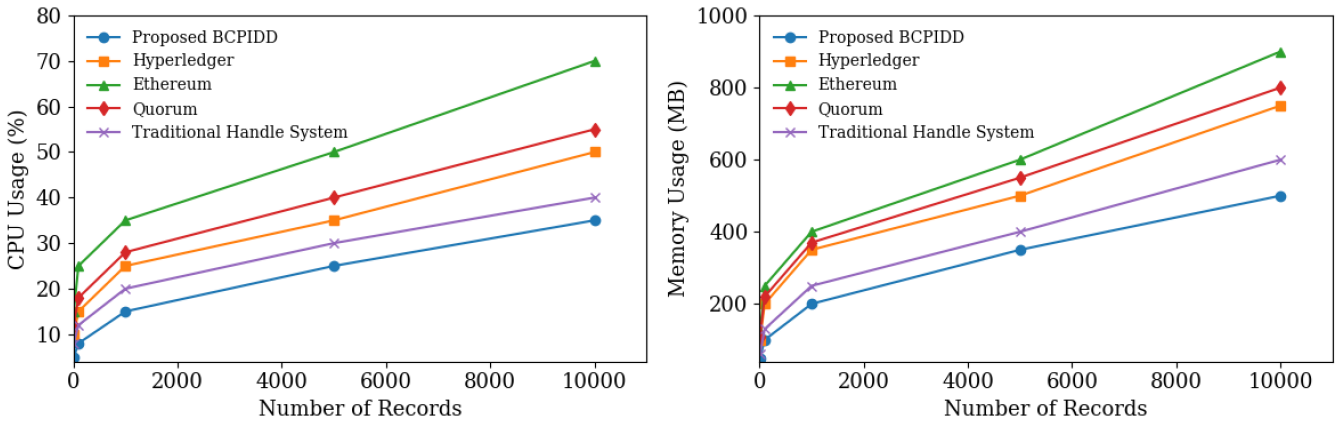

Fig. 10. Resources usage

C. Scalability and Stress Testing

In this section, a test is carried out to find the robustness of the proposed system by examining it beyond the limits of normal operation.

1) CPU and Memory Utilization over 24 Hours : We analyzed the CPU and memory utilization of the system over a 24-hour period, showcasing dynamic fluctuations and peak loads. Fig.11. shows the analysis of the observed trends. Comparative analysis of CPU and memory utilization over 24 hours reveals critical insights into the robustness and adaptability of the systems.

The BCPIDD system demonstrated exceptional stability and efficiency over time, with CPU utilization fluctuating modestly between 30% and 40%, even during peak loads. This consistency highlights the system's optimized use of Tendermint consensus and decentralized IPFS storage, which effectively distribute processing demands. By contrast, Hyperledger Fabric exhibited significant spikes in CPU usage, ranging from 45% to 65%, reflecting the computational overhead of its endorsement and validation mechanisms. Similarly, Ethereum showed the highest variability in CPU usage, peaking at nearly 80% during high-demand periods, due to its reliance on resource-intensive block validation processes. Quorum, while more stable than Ethereum, demonstrated intermediate performance with fluctuations between 50% and 65%, indicative of its modified consensus design. The Handle System, with its centralized architecture, maintained CPU usage in the range of 35% to 45%, benefiting from its simplicity but lacking the scalability of distributed systems.

Memory utilization further emphasizes the differences between these systems. BCPIDD maintained a consistent and relatively low memory footprint, fluctuating between 480 MB and 520 MB, underscoring its efficient data storage and retrieval design. Hyperledger Fabric and Ethereum, on the other hand, required significantly more memory, with peak utilization reaching 800 MB and 950 MB, respectively. These demands arise from their complex transaction processing and storage replication requirements. Quorum exhibited moderate memory usage, varying between 750 MB and 850 MB, while the Handle System displayed the lowest variability, ranging from 580 MB to 620 MB, due to its centralized nature and absence of redundancy mechanisms.

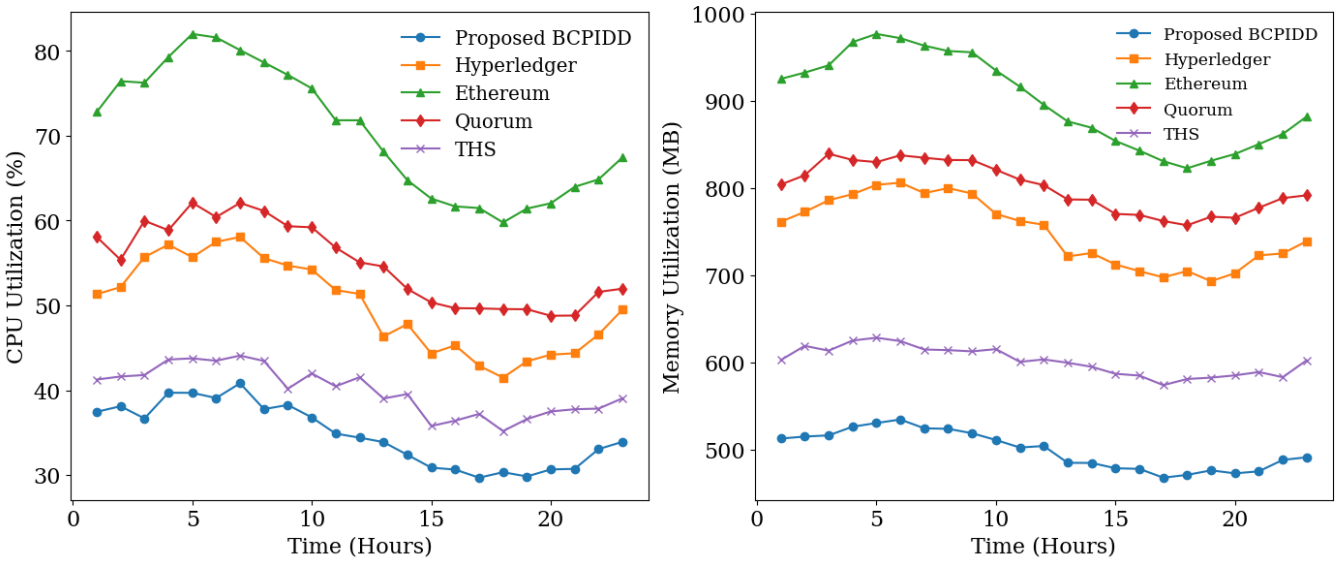

Fig. 11. Resources usage over 24 hours

2) Operation under Extreme Disruptions: The resilience of systems under node failure conditions is critical for assessing their robustness and operational reliability. Simulating a 50% node failure period provided valuable insights into the performance degradation and recovery capabilities of BCPIDD, Hyperledger Fabric, Ethereum, Quorum, and the Traditional Handle System.

• Latency During 50% node failure

Latency was observed to increase across all systems during the failure period, with BCPIDD demonstrating the smallest relative increase. Fig.12 shows the latency during 50% of nodes failures.

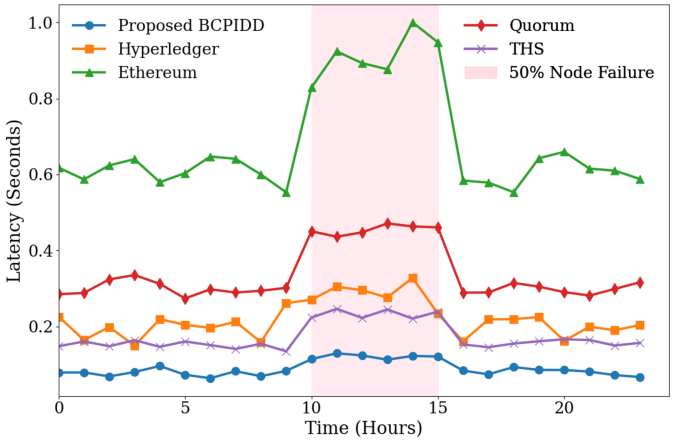

Fig. 12. Latency under 50% node failure

BCPIDD's latency rose by approximately 50%, maintaining an upper limit of 0.12 seconds even under node failure. This highlights the system's efficient consensus mechanism and decentralized architecture, which distribute the computational load effectively. In contrast, Hyperledger Fabric and Ethereum experienced significant latency increases, with Ethereum exceeding 1.05 seconds during the disruption. This performance degradation can be attributed to the reliance on computationally intensive block validation processes and increased overhead during node reorganization. The Handle System, although less affected, showed an increase to 0.30 seconds, reflecting its centralized nature and reliance on a single point of resolution.

• Throughput During 50% node failure

When a node fails, the system's overall capacity to process data and perform tasks is reduced. This can lead to a drop in throughput, which is the amount of data that can be processed or transmitted over a certain period. Throughput declined significantly for all systems during the node failure, with reductions of up to 50% as expected. Fig.13. shows an illustration of throughput during 50% node failure. BCPIDD maintained the highest throughput during the disruption, at approximately 10,800 TPS, reflecting its ability to sustain transaction processing despite adverse conditions. Hyperledger Fabric and Quorum, while still functional, exhibited reductions to 9,600 TPS and 440 TPS, respectively, highlighting their susceptibility to network disruptions and increased load on remaining nodes. Ethereum, with its inherently lower throughput, experienced the most severe impact, dropping to 400 TPS. The Handle System, while not decentralized, maintained a throughput of 2,300 TPS, indicating its lower complexity and reduced dependence on network stability.

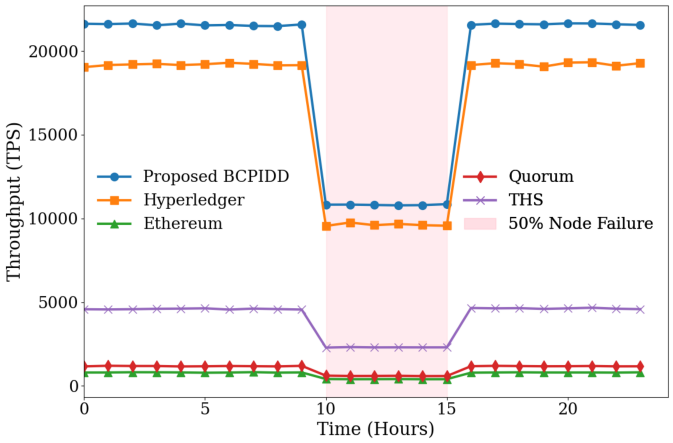

Fig. 13. Throughput under 50% node failure

• Response Time During 50% Failure

The response time which is critical for user-facing applications, displayed notable increases during the failure period. BCPIDD's response time doubled to 0.28 seconds, maintaining its position as the most responsive system under failure conditions (See Fig.14 for illustration). Hyperledger Fabric and Ethereum experienced response time increases to 0.52 seconds and 1.16 seconds, respectively, reflecting the increased computational and network overhead during failure recovery. The Handle System's response time rose moderately to 0.64 seconds, again reflecting its simpler architecture and centralized resolution model.

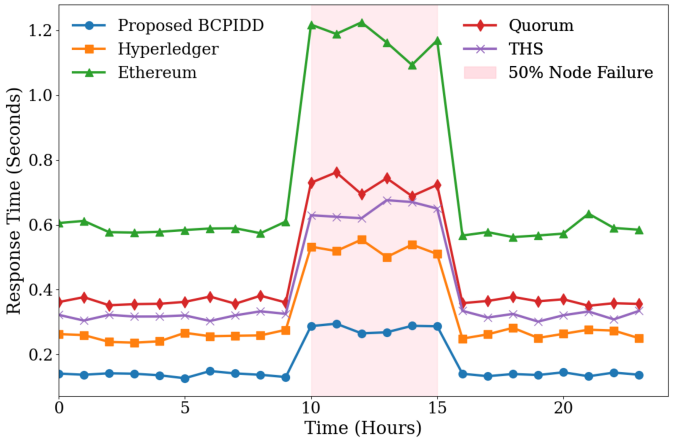

Fig. 14. Response time under 50% node failure

D. Security Analysis

BCPIDD security was evaluated under adversarial conditions, focusing on its resilience against Distributed Denial of Service (DDoS) attack data tampering. The results demonstrate its robustness compared to the traditional Handle System, Ethereum, Hyperledger Fabric, and Quorum.

• Latency During Attack

The simulation of latency during DDoS and data tampering attacks reveals the distinct resilience and vulnerabilities of the evaluated systems. Blockchain-based systems, such as BCPIDD, Hyperledger Fabric, Ethereum, and Quorum, demonstrate their efficiency by maintaining operational latency under both attack scenarios. In contrast, the centralized

Traditional Handle System fails completely, highlighting the inherent weaknesses of centralized architectures. An illustration is provided by the Fig.15.

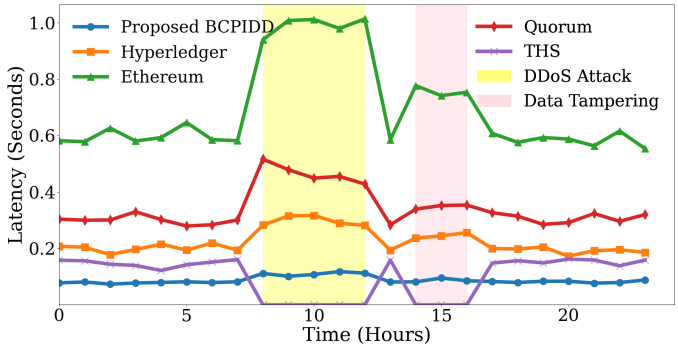

Fig. 15. Latency under attacks

The latency of BCPIDD rises from a baseline of approximately 0.08 seconds to about 0.104 seconds, a 30% increase. Hyperledger Fabric shows a similar increase, with latency climbing from 0.2 seconds to approximately 0.26 seconds. Ethereum and Quorum, which have higher baseline latency due to their consensus mechanisms, experiences comparable proportional increases. These findings align with the expected behavior of blockchain systems, where decentralized consensus and cryptographic validation introduce delays under stress. The Traditional Handle System, however, drops to zero functionality during the DDoS attack, with no latency recorded. This reflects the system's complete failure to respond to requests, as its centralized architecture leaves it vulnerable to overwhelming traffic directed at a single point of failure. This inability to sustain operations under attack highlights the critical limitations of centralized designs in high-risk environments.

Under the data tampering scenario, blockchain systems again exhibit resilience. The increase in latency during this period is less pronounced, with BCPIDD's latency rising by about 10% to 0.088 seconds, while Hyperledger's latency increases to 0.22 seconds. These minor delays are attributed to the additional validation required to detect and reject tampered data, a fundamental feature of blockchain's integrity-preserving design. Ethereum and Quorum similarly maintain functionality, with proportional latency increases consistent with their architectural constraints. The Handle System, as in the DDoS attack, fails entirely during data tampering. Its inability to validate data or detect unauthorized modifications results in a complete operational breakdown. This outcome underscores the system's unsuitability for environments requiring tamper-resistant solutions, as even minimal malicious activity can compromise its functionality.

• Throughput During Attack

The throughput results during DDoS and data tampering attacks highlight significant differences in resilience between blockchain-based systems and the centralized Traditional Handle System. Blockchain systems such as BCPIDD, Hyperledger Fabric, Ethereum, and Quorum demonstrate varying levels of robustness, while the Handle System completely fails during both attack scenarios(As shown in Fig. 16). During the DDoS attack, blockchain systems experienced moderate reductions in throughput. For example, BCPIDD's

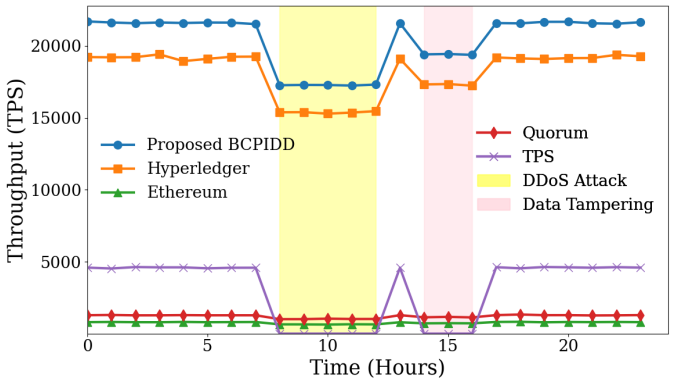

Fig. 16. Throughput during attacks

throughput decreased by approximately 20%, from its baseline of 21,600 TPS to around 17,280 TPS, reflecting the network's ability to maintain operations despite increased traffic. Similarly, Hyperledger Fabric exhibited a similar reduction, maintaining roughly 15,360 TPS during the attack compared to its normal throughput of 19,200 TPS. These reductions are indicative of the systems' resilience under stress, as the decentralized architecture distributes the load and mitigates the impact on individual nodes. Ethereum and Quorum, with lower baseline throughput of 800 TPS and 880 TPS respectively, also maintained functionality but showed proportional decreases consistent with their architectural limitations. Conversely, the Traditional Handle System was completely overwhelmed by the DDoS attack, with throughput dropping to zero. This outcome underscores the vulnerability of centralized systems to targeted attacks, as the single point of failure in such architectures renders them unable to handle large volumes of malicious traffic.

In the data tampering scenario, blockchain systems once again demonstrated their robustness. Throughput reductions were less pronounced compared to the DDoS attack, as the tampering primarily tested the systems' ability to detect and reject malicious data. BCPIDD maintained approximately 19,440 TPS during this period, a decrease of only 10% from its baseline. Similarly, Hyperledger Fabric showed a modest reduction to 17,280 TPS, reflecting the computational overhead of cryptographic validation and consensus. Ethereum and Quorum exhibited comparable behavior, with proportional throughput decreases while remaining operational. The Handle System, however, failed entirely during data tampering, with throughput again dropping to zero. The system's inability to validate data or detect unauthorized modifications led to a complete operational breakdown, highlighting its fundamental design limitations in the context of tamper resistance.

• Response Time During Attack

The response time results during DDoS and data tampering attacks provide key insights into the performance and resilience of blockchain-based systems compared to the centralized Traditional Handle System. Blockchain systems, including BCPIDD, Hyperledger Fabric, Ethereum, and Quorum, demonstrate an ability to maintain operations under stress, albeit with some increases in response time. In contrast, the Handle System fails entirely during both types of attacks, underscoring its vulnerabilities, it can be

seen in Fig.17. During the DDoS attack, blockchain systems exhibit a moderate increase in response time due to the computational overhead required to manage the surge in network traffic. For instance, BCPIDD's response time rises from its baseline of approximately 0.14 seconds to 0.21 seconds, representing a 50% increase. Hyperledger Fabric shows a similar trend, with response times increasing from 0.26 seconds to around 0.39 seconds. Ethereum and Quorum, which have higher baseline response times due to their consensus algorithms, also exhibit proportional increases during the attack. These increases highlight the added burden placed on blockchain systems to validate transactions and maintain consensus under high-load conditions, yet they remain operational and responsive. In stark contrast, the Traditional Handle System's response time drops to zero during the DDoS attack, reflecting a complete breakdown in its ability to handle incoming requests. The data tampering

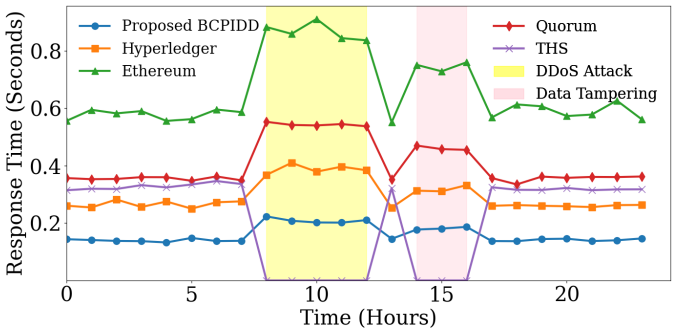

Fig. 17. Response time during attacks

scenario further emphasizes the robustness of blockchain systems. While response times increase slightly as the systems validate and reject tampered data, the increases are minimal compared to the DDoS attack. BCPIDD's response time rises to approximately 0.18 seconds, an increase of only 30%. Hyperledger Fabric and Ethereum exhibit similar increases, maintaining response times well within acceptable limits despite the additional computational workload. This ability to detect and handle tampered data without significant performance degradation underscores the resilience of blockchain systems, which rely on cryptographic hashing and decentralized validation to ensure data integrity. The Handle System, once again, fails completely during data tampering, with response times dropping to zero. This total collapse reflects its inability to validate or process transactions in the presence of unauthorized modifications. The lack of robust validation mechanisms and redundancy in the Handle System makes it unsuitable for environments where tamper resistance is critical.

The results highlight the critical differences in resilience and functionality between decentralized blockchain systems and centralized architectures. Blockchain-based credential are able to maintain responsiveness under both DDoS and data tampering attacks, demonstrating their suitability for high-security, high-reliability applications.

V. Conclusion

In this paper, we proposed the Blockchain-based Persistent Identifier for the diplomas system to enhance the capability of the Handle System in managing digital diplomas with low vulnerabilities such as the single point of failure. BCPIDD integrates blockchain, using Tendermint for fast and secure consensus, Merkle Trees for ensuring data integrity, and IPFS for decentralized storage. The experiment conducted in AWS environments shows that BCPIDD outperforms the traditional Handle System, and also existing blockchain systems, including Hyperledger Fabric, Ethereum, and Quorum. It showed better resilience under extreme disruptions, as well as under DDoS attacks and data tampering. These results suggest that a transition to decentralized, blockchain-powered solutions is essential for managing sensitive and high-value data in increasingly hostile digital environments.

In future work, it will be interesting to explore the integration of other techniques to enhance the BCPIDD performance in the prediction of transaction load and dynamic adjustment of resources. And integrate machine learning with the persistent identifiers, to predict the eventual alterations. Additionally provides support for more types of digital credentials with increased interoperability within existing educational systems. Another potential work to explore is the application of persistent identifiers in healthcare, to manage the patients' data. Those identifiers can make medical information more accessible, operable, and reusable.

Acknowledgment

The authors would like to thank Wuxi Handle Company for making their laboratory available for experiments.

References

[1] D. Nadrljanski, M. Nadrljanski, and M. Pavlinović, *Digitalization of Education*. Cham: Springer International Publishing, 2022, pp. 17–39.
[2] P. Parviainen, M. Tihinen, J. Kääriäinen, and S. Teppola, "Tackling the digitalization challenge: how to benefit from digitalization in practice," *International Journal of Information Systems and Project Management*, vol. 5, no. 1, p. 63–77, Feb. 2022.
[3] J. Vrana and R. Singh, *Digitization, Digitalization, and Digital Transformation*. Cham: Springer International Publishing, 2021, pp. 1–17.
[4] D. S. Marathe, "Digitalization in education sector," *International Journal of Trend in Scientific Research and Development*, vol. Special Issue, pp. 51–56, 10 2018.
[5] G. W. Matkin, *The Challenge of Digital Credentials: How Should Universities Respond?* Singapore: Springer Singapore, 2020, pp. 51–61.
[6] M. A. Hisseine, D. Chen, X. Yang, and X. Wang, "Using handle system to provide persistent identifiers for diploma," 2022, pp. 68–73.
[7] C. Barabas and P. Schmidt, "Transforming chaos into clarity: The promises and challenges of digital credentialing," *The Roosevelt Institute*, 2016.
[8] A. El Koshiry, E. Eliwa, T. Abd El-Hafeez, and M. Y. Shams, "Unlocking the power of blockchain in education: An overview of innovations and outcomes," *Blockchain: Research and Applications*, vol. 4, no. 4, p. 100165, 2023.
[9] C. Pujari, C. CB, S. R. Muppidi, and M. C. Belavagi, "A novel method of secure child adoption using blockchain technology," *IAENG International Journal of Applied Mathematics*, vol. 53, no. 4, pp. 1531–1539, 2023.
[10] P. Liu, B. Zhao, and Q. Liu, "Blockchain technology investment strategies of green agri-food supply chain under government tax subsidy strategy." *Engineering Letters*, vol. 32, no. 5, pp. 949–964, 2024.
[11] A. Mohammad and S. Vargas, "Challenges of using blockchain in the education sector: A literature review," *Applied Sciences*, vol. 12, no. 13, 2022.
[12] P. S. Gharat, G. Choudhary, S. K. Shandilya, and V. Sihag, "Future applications of blockchain in education sector: A semantic review," in *Mobile Internet Security*, I. You, H. Kim, T.-Y. Youn, F. Palmieri, and I. Kotenko, Eds. Singapore: Springer Nature Singapore, 2022, pp. 93–106.
[13] M. N. Abdullaht, "A survey on blockchain-based applications in education," in *2020 7th International Conference on Computing for Sustainable Global Development (INDIACom)*, 2020, pp. 266–270.

[14] "Ethereum: A secure decentralised generalised transaction ledger," 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:261284443
[15] X. Zheng, J. Lu, S. Sun, and D. Kiritsis, "Decentralized industrial iot data management based on blockchain and ipfs," in *Advances in Production Management Systems. Towards Smart and Digital Manufacturing*, B. Lalic, V. Majstorovic, U. Marjanovic, G. von Cieminski, and D. Romero, Eds. Cham: Springer International Publishing, 2020, pp. 222–229.
[16] V. Garcia-Font, *Blockchain: Opportunities and Challenges in the Educational Context*. Cham: Springer International Publishing, 2020, pp. 133–157.
[17] V. F. Richter, G. Feliz, and S. Christophe, "Revocation mechanisms for academic certificates stored on a blockchain," in *2020 15th Iberian Conference on Information Systems and Technologies (CISTI)*, pp. 1–6.
[18] G. Capece, N. L. Ghiron, and F. Pasquale, "Blockchain technology: Redefining trust for digital certificates," *Sustainability*, vol. 12, 2020.
[19] ORCID, "Orcid and persistent identifiers."
[20] S. Sun, *Establishing Persistent Identity using the Handle System.*, 1 2001.
[21] C. of Handle Services (China), "Handle system introduction," 2019.
[22] M. A. Hisseine, D. Chen, Y. Xiao, and P. K. Alimo, "A review of digital object architecture and handle system: Development, current applications and prospective," *Internet of Things*, vol. 26, p. 101230, 2024.
[23] J. You and X. Li, "Optimization and implementation of handle server," 2014, pp. 682–686.
[24] L. Shiqiang, Z. Xinyi, and Q. Jiacai, "Introduction to handle system," *China CIO News*, vol. 312, pp. 38–39, 2019.
[25] G. Xiaofeng and X. S. Sam, "The development and application of handle system," *Digital library forum*, vol. 8, pp. 18–24, 2013.
[26] A. Durand, "Digital object architecture and the handle system," 2019.
[27] S. Sun, L. Larry, and B. Brian, "Rfc3650: Handle system overview," 2003. [Online]. Available: https://www.rfc-editor.org/rfc/rfc3650
[28] X. S. Sam, R. Sean, and L. un Larry, "Handle system namespace and service definition," pp. 1–41, 11 2003.
[29] P. Maniriho and T. Ahmad, "Enhancing the capability of data hiding method based on reduced difference expansion," *Engineering Letters*, vol. 26, no. 1, pp. 45–55, 2018.
[30] BCdiploma, "The premier credentialing & badging experience," 2024.
[31] A. Taylor and R. Parks, "Blockchain: A future technology for registrars?" *The Successful Registrar*, vol. 20, pp. 1–4, 2020.
[32] W. Rojas, V. Gayoso Martínez, and A. Queiruga-Dios, "Blockchain in education: New challenges," in *13th International Conference on Computational Intelligence in Security for Information Systems (CISIS 2020)*, Á. Herrero, C. Cambra, D. Urda, J. Sedano, H. Quintián, and E. Corchado, Eds. Cham: Springer International Publishing, 2021, pp. 380–389.
[33] H. A. M. Deenmahomed, M. M. Didier, and R. K. Sungkur, "The future of university education: Examination, transcript, and certificate system using blockchain," *Computer Applications in Engineering Education*, vol. 29, no. 5, pp. 1234–1256, 2021.
[34] M. Turkanović, M. Holbl, K. Ko, M. Heriko, and A. Kamialić, "Eductx: A blockchain-based higher education credit platform," *IEEE Access*, vol. 6, pp. 5112–5127, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:3404334
[35] A. Gayathiri, J. Jayachitra, and S. Matilda, "Certificate validation using blockchain," in *2020 7th International Conference on Smart Structures and Systems (ICSSS)*, 2020, pp. 1–4.
[36] D. S. V. Madala, M. P. Jhanwar, and A. Chattopadhyay, "Certificate transparency using blockchain," in *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, 2018, pp. 71–80.
[37] Y. C. Elloh Adja, B. Hammi, A. Serhrouchni, and S. Zeadally, "A blockchain-based certificate revocation management and status verification system," *Computers & Security*, vol. 104, p. 102209, 2021.
[38] M. Caramihai and I. Severin, "A blockchain-based solution for diploma management in universities," *Sustainability*, vol. 15, no. 20, 2023. [Online]. Available: https://www.mdpi.com/2071-1050/15/20/15169
[39] Q. Tang, "Towards using blockchain technology to prevent diploma fraud," *IEEE Access*, vol. 9, pp. 168 678–168 688, 2021.
[40] I. E. Khairuddin, N. A. Safian, M. K. Zaini, and N. A. Uzir, "Navigating academic credentials in the digital age: Insights, challenges, and implications for graduates," *Journal of Ecohumanism*, vol. 3, no. 4, p. 642–651, Jul. 2024.
[41] G. Bjelobaba, M. Paunovic, A. Savic, H. Stefanovic, J. Doganjic, and Z. Miladinovic Bogavac, "Blockchain technologies and digitalization in function of student work evaluation," *Sustainability*, vol. 14, no. 9, 2022.
[42] A. Rustemi, V. Atanasovski, A. Risteski, and B. Popovski, "Analysis of blockchain platforms for generation and verification of diplomas," in *2023 12th Mediterranean Conference on Embedded Computing (MECO)*, 2023, pp. 1–4.