Ingénierie des Systèmes d'Information
Vol. 30, No. 5, May, 2025, pp. 1297-1311
Journal homepage: http://iieta.org/journals/isi

Quantum-Resistant Identity Management via ZK-STARKs and Decentralized Storage

Khalid Maidine[1*], Ahmed EL-Yahyaoui[1], Salima Trichni[1;2]

[1] Intelligent Processing and Security of Systems, Faculty of sciences, Mohammed V University in Rabat, Rabat 10100, Morocco
[2] Department of Interdisciplinary Modules, Faculty of Economics, Legal and Social Sciences of Sale, Mohammed V University in Rabat, Rabat 10100, Morocco

Corresponding Author Email: khalid_maidine@um5.ac.ma

https://doi.org/10.18280/isi.300516

Received: 19 March 2025
Revised: 12 May 2025
Accepted: 22 May 2025
Available online: 31 May 2025

Keywords:
blockchain technology, IPFS, identity management, quantum computing, smart contract, ZK-STARK, decentralized identity, privacy-preserving technologies

ABSTRACT

Traditional identity management systems are inherently vulnerable to critical issues, including pervasive privacy breaches and single points of failure, which compromise the security and integrity of sensitive user information. These centralized models require the disclosure of personal data to third parties, thereby increasing the risk of misuse, exploitation, and large-scale data leaks. To address these limitations alongside the emerging threat posed by quantum computing this paper proposes a novel identity management architecture that integrates Zero-Knowledge Scalable Transparent Arguments of Knowledge, the InterPlanetary File System, and blockchain technology. This design enables decentralized, privacy-preserving identity verification by allowing users to prove specific identity attributes without revealing the underlying sensitive data, and without the need for trusted third parties. The use of the InterPlanetary File System ensures that encrypted user data is stored off-chain in a distributed, immutable manner, reducing exposure risks and enhancing availability. A functional prototype of the system was developed using a Zero-Knowledge Scalable Transparent Arguments of Knowledge cryptographic library and evaluated to demonstrate its practical feasibility. The evaluation confirms that the architecture is efficient, scalable, and resistant to quantum attacks, making it a strong candidate for real-world digital identity systems. This work provides a forward-looking, secure, and privacy-preserving alternative to traditional identity frameworks.

1. INTRODUCTION

1.1 Background

The proliferation of online services, while delivering unprecedented user convenience, has simultaneously exposed critical vulnerabilities within the foundational identity protocols such as OAuth, OIDC, and SAML that underpin the digital ecosystem [1]. These systems, which rely heavily on centralized authorities [2], are particularly susceptible to cyberattacks because they store vast amounts of sensitive user data in one central location. This creates a "honeypot" effect, making them attractive targets for cybercriminals. Once breached, these systems can expose the personal information of millions (Table 1), leading to serious privacy and security issues.

Several recent incidents in 2025 demonstrate the extent of vulnerabilities leading to the exposure of sensitive user information. Cybercriminals continue to target organizations across various sectors, successfully exfiltrating personal data such as names, addresses, Social Security numbers, dates of birth, and phone numbers.

For example, in May 2025, a massive data exposure, dubbed the "Mega-Leak," came to light, affecting an estimated 184.1 million user credentials [3]. This breach involved an unsecured online database containing plaintext usernames, passwords, and login links for major platforms including Google, Apple, Microsoft, Facebook, Instagram, and Snapchat. The exposed data also included login details for banking and financial institutions, healthcare services, and even government portals, with some. gov email addresses found within the dataset. Cybersecurity researcher Jeremiah Fowler, who discovered the database, believes the data originated from widespread infostealer malware campaigns designed to siphon sensitive information like browser logins, cookies, and autofill data from infected devices. The direct availability of such a vast trove of plaintext credentials poses an extreme risk of identity theft, financial fraud, and further targeted attacks.

The Canadian utility provider Emera Power/Nova Scotia Power reported in April/May 2025 that a ransomware attack, which began around March 19, 2025, and was detected on April 25, 2025, resulted in the theft of personal and financial data belonging to approximately 280,000 customers [4]. The stolen information, which included names, phone numbers, email addresses, mailing addresses, dates of birth, account history, driver's license numbers, Social Insurance Numbers (SINs), and bank account numbers, was subsequently published on the dark web. The company confirmed it did not pay the ransom demand.

Table 1. Major data breaches in 2025

Organization	Date	Data Records	Type of Data	Mitigation
"Mega-Leak" (Google, Apple, Microsoft, Facebook, etc.)	May 2025	184.1 million	Usernames, passwords, email addresses, login links (including for bank, health, government portals)	Eliminates central data "honeypots" with decentralized IPFS storage. Replaces vulnerable passwords with secure wallet-based authentication, preventing password theft.
LexisNexis Risk Solutions	Apr 2025	364,333	Names, phone numbers, postal and email addresses, Social Security numbers, driver's license numbers, dates of birth	Stores data decentrally on IPFS, preventing exposure from a single compromised platform. Uses zk-STARKs for verification, so raw personal data is never shared or exposed.
Emera Power / Nova Scotia Power	May 2025	280	Names, phone numbers, email addresses, mailing addresses, dates of birth, driver's license numbers, Social Insurance Numbers, bank account numbers	Removes the central target for ransomware via decentralized storage. User-controlled encryption makes stolen data files unusable to attackers.
Adidas	May 2025	Not Specified	Names, email addresses, phone numbers, physical addresses	Reduces third-party risk by using zk-STARK proofs for verification, so vendors never hold or store raw user data. Users maintain direct control over their information.
Select Medical Holdings	June 2025	Not Specified	Names, dates of birth, addresses, provider names, dates of service, patient account numbers, Social Security numbers (in some cases)	Prevents vendor data breaches by storing patient data decentrally under user control. Uses zk-STARKs to verify information without sharing sensitive data like SSNs.
TeleMessage	May 2025	Not Specified	Names, message fragments, contact information of US government personnel	Enforces user-side encryption before data is stored, so service providers cannot access unencrypted information. Secures identity with cryptographic wallets instead of vulnerable service accounts.

Sportswear giant Adidas confirmed a data breach in May 2025 where customer contact information was compromised via an attack on a third-party customer service provider [5]. The exposed data included names, email addresses, phone numbers, and physical addresses. Adidas stated that sensitive financial data, such as payment card details and passwords, was not affected.

Select Medical Holdings notified patients in June 2025 about a data security incident impacting its former vendor, Nationwide Recovery Services, Inc [6]. The breach at the vendor was detected on July 11, 2024, and a review completed by February 3, 2025, confirmed patient data was accessed. Exposed information potentially included names, dates of birth, addresses, provider names, dates of service, patient account numbers, and, in some cases, Social Security numbers.

TeleMessage, a company providing a modified version of the Signal messaging app for regulatory archiving, was reportedly hacked twice in May 2025 [7]. The breach exposed administrator credentials and unencrypted message content, including names, message fragments, and contact information associated with users from U.S. government agencies.

LexisNexis Risk Solutions disclosed in April 2025 that a data breach occurring on December 25, 2024, had compromised the personal information of 364,333 individuals [8]. An unauthorized third party gained access to the company's GitHub account by exploiting a vulnerability in a third-party software platform used for software development. The stolen data could include names, phone numbers, postal and email addresses, Social Security numbers, driver's license numbers, and dates of birth.

Whether through exploiting unpatched software vulnerabilities, misconfigurations, or social engineering attacks like phishing, cybercriminals are able to penetrate these systems with alarming ease. The reliance on a single point of failure means that once a system is breached, the consequences are widespread, affecting millions of individuals and potentially causing long-term damage to the organization’s reputation and financial stability.

1.2 Motivation

The repeated occurrence of high-profile data breaches underscores the critical need for more secure and decentralized identity management systems. Centralized platforms, where personal data is stored in a single location, are particularly vulnerable to exploitation through cyberattacks and human error [2]. These breaches not only lead to massive leaks of sensitive information but also undermine public trust, making users reluctant to share their personal data online. The impact is profound: digital services are increasingly seen as unsafe, slowing the adoption of innovative technologies.

To address this issue, decentralized solutions, such as those built on blockchain technology [9], offer a promising alternative. By decentralizing data storage and verification, these systems eliminate single points of failure and provide greater security. For instance, the InterPlanetary File System (IPFS) could be integrated into identity management frameworks, allowing data to be stored across a distributed network rather than on a single server. This would make it harder for attackers to locate and exploit sensitive information.

In addition, advanced cryptographic techniques [10] like Zero-Knowledge Proofs can enhance privacy within these decentralized systems. ZKPs enable users to verify their identity without actually revealing their personal data, adding a further layer of protection against data breaches. This combination of blockchain, IPFS, and ZKP can result in a robust identity management system that reduces reliance on centralized control, improves security, and mitigates risks associated with quantum computing.

Moreover, IPFS [11] can ensure that personal data remains encrypted and fragmented across various nodes, reducing the chances of mass data exposure. It also aligns well with future-proofing efforts against quantum threats, which could potentially break traditional encryption methods. By integrating blockchain with IPFS and ZKP, identity management can evolve into a more secure, transparent, and resilient framework.

By integrating these technologies Blockchain, ZKP and IPFS, we can build systems that not only improve security and reduce reliance on centralized control but also prepare for future threats, such as quantum computing [12].

1.3 Contribution

This paper proposes a new and innovative framework for identity management that incorporates blockchain technology, ZK-STARK, and IPFS in one unified system to solve the triad problems of security, privacy, and availability in digital identity management. Moreover, the architecture described in this paper shifts toward better user information protection from centralized systems that store data on servers because they require higher levels of security. In addition, due to blockchains decentralized nature as well as its immutability feature, the framework has eliminated weak spots related to single points of failure which is critical issue towards providing strong underlying infrastructure for auditable identity management. Also, the integration of the IPFS makes it possible to store user data in a distributed manner while maintaining redundancy which supports off-chain storage with guaranteed accessibility and integrity. Another contribution is use of ZK-STARK , which enable private attestations without disclosing sensitive information， and age verification attesting where users can prove being older than a threshold age without disclosing full date of birth at which age calculation was performed. This prototype auspiciously attests system practicality and demonstrates combining zero-knowledge proofs with decentralized storages effectiveness toward privacy-preserving identity verification systems. Additionally, this framework approaches quantum.

2. RELATED WORKS

The evolution of digital identity management has been marked by a significant trend away from centralized platforms toward decentralized, blockchain-based systems designed to enhance user security and control. Initial implementations of blockchain-based identity, such as Blockchain-based Identity as a Service (BIDaaS) [13], demonstrated the feasibility of decentralizing user authentication. However, these early models were limited in their decentralization, often retaining a central entity for managing identity attributes, which reintroduced risks associated with a single point of failure and did not fully resolve privacy concerns. This underscored the necessity for more advanced systems capable of both decentralizing control and fundamentally protecting user privacy during verification processes.

To address these privacy deficiencies, subsequent research integrated Zero-Knowledge Proofs, which enable information verification without exposing the underlying data. A significant portion of this research utilized zk-SNARKs to secure identity transactions [14]. This approach was applied across various domains, including frameworks for digital identity management on the blockchain, privacy-preserving healthcare credentials like vaccination [15] passes, and the securing of electronic health records (EHRs) [16]. Other proposed systems combined zk-SNARKs with the Ethereum blockchain and the IPFS to construct Self-Sovereign Identity frameworks [17].

Another approach to attribute privacy involves specialized cryptographic methods like range proofs, which verify that a value falls within an interval without revealing the specific value. Addressing this, the study [18] developed an efficient DID system for social networks using a novel range proof protocol based on Pointcheval-Sanders (PS) signatures.

Despite their utility, zk-SNARKs possess inherent limitations that pose considerable challenges to their widespread adoption. A primary vulnerability is the requirement of a "trusted setup," an initial parameter-generation event that, if compromised, could undermine the security of the entire system. Furthermore, the computational overhead associated with generating zk-SNARK proofs can impede scalability, especially in high-throughput applications. Critically, the cryptographic principles that zk-SNARKs are built upon, such as elliptic curves and pairings, are vulnerable to cryptanalytic attacks from future quantum computers.

In response to these challenges, zk-STARKs have emerged as a more robust and forward-looking alternative. A defining feature of zk-STARKs is that they do not require a trusted setup, which provides full transparency and eliminates a significant security risk. They are constructed using hash functions, rendering them inherently resistant to quantum attacks and ensuring long-term cryptographic integrity. Moreover, zk-STARKs offer superior scalability and efficiency for complex computations, making them highly suitable for large-scale identity management systems. While comparative studies have highlighted zk-STARKs as a promising alternative to zk-SNARKs, a key research gap persists in their practical application.

Although zk-STARKs have been theoretically explored in identity systems [19], our framework goes further by implementing them in a functioning prototype with Ethereum and IPFS integration. The challenges associated with zk-STARKs' larger proof sizes and the need for their efficient pairing with off-chain storage have remained underdeveloped in existing literature. This paper seeks to address this gap by proposing a novel architecture that synthesizes zk-STARKs, IPFS, and blockchain technology. By leveraging the unique advantages of each component, our work presents a solution designed to overcome the privacy, security, and scalability limitations of prior identity management systems.

3. PRELIMINARIES

3.1 Blockchain

Since the publication of Bitcoin’s whitepaper in 2008 [20], followed by the launch of Bitcoin in 2009 [21], cryptocurrencies have significantly influenced traditional finance. At the heart of Bitcoin lies blockchain, a distributed ledger technology. In this system, peers connect individual blocks in a chronological sequence, forming a secure data structure that ensures immutability and integrity through cryptographic methods. Because blockchain transactions do not require intermediaries, they are transparent, traceable, and resistant to tampering. This decentralized nature establishes a strong trust mechanism in an environment that does not rely on central authorities. Blockchains can be categorized based on whether they require permission for peers to join or leave the network. Permissionless [22] or simply public blockchains, are fully decentralized networks where transactions and incentives are verified through consensus mechanisms involving unknown participants. This structure supports a decentralized model of trust, where all participants have equal

access to the network without the need for authorization. In contrast, permissioned blockchains [23] include consortium and private blockchains. Consortium blockchains are managed by a group of institutions that collectively decide the level of access and openness to the public, depending on the use case. These blockchains typically use alternative consensus mechanisms, such as Proof of Stake (PoS) or Practical Byzantine Fault Tolerance (PBFT), instead of the energy-intensive Proof of Work (PoW). Private blockchains, on the other hand, restrict access to a single entity or a select group, making them suitable for scenarios with a limited number of participants. While this structure enhances control and security, it results in a more centralized system with a narrower scope of application compared to public blockchains.

3.2 Smart contracts

Smart contracts are programs that operate themselves in a decentralized manner, allowing the execution of agreements without the intervention of intermediaries, usually executed in blockchains like Ethereum [24]. They are used in various applications, such as financial transactions and identity management, where they automate tasks like verifying identity attributes (e.g., citizenship or educational credentials) based on predefined rules. For instance, a smart contract can validate a user's citizenship status by confirming a valid passport without exposing unnecessary sensitive information, improving reliability and reducing the chance of human error. Furthermore, smart contracts dynamically manage Decentralized Identifiers, automatically updating or revoking credentials like expired professional certifications, ensuring only valid records are recognized. Latest improvements in contract security and auditing may become a turning point for smart contracts as vulnerability events, like the DAO attack in 2016 [25], can bring them down completely. Techniques like formal verification, symbolic execution, and fuzzing have improved their reliability. Formal verification mathematically ensures contracts meet specifications, while symbolic execution explores potential vulnerabilities by examining all execution paths. Fuzzing [26, 27], since 2018, has become a key tool in discovering unexpected vulnerabilities by generating random inputs. The main focus of future improvements of these techniques lies on better performance, more effective bug detection (test oracles), as well as high-quality initial inputs, leading to safer and more efficient smart contracts.

3.3 Decentralized identifiers

DIDs represent a major innovation in blockchain-based identity management that open a self-sovereign identity space limitless by traditional identifiers usually monitored by centralized entities [17]. They are fundamentally different from traditional ones, which are operated by third-party authorities such as governments or corporations, as DIDs give users the power to create and manage their own identifiers without anyone's help. Such a model removes the necessity to trust intermediaries, allowing people to get full ownership and control of their identity credentials. These identifiers, which are securely saved on blockchain, utilize smart contracts to confirm the truth of the credentials that are given, thus the process of verification is both smooth and safe. The decentralized character of DIDs not only reduces the chances of single points of failure which are a major weakness in centralized systems but also improve privacy and data security for users in the digital ecosystem [28].

3.4 ZK-Starks

ZK-STARKs were introduced in 2018 [29] as a groundbreaking proof system for cryptography. The systems were developed by Eli Ben-Sasson and his group, ZK-STARKs being attempts to improve the performance and handling of security issues of the zero-knowledge proof systems, for example, ZK-SNARKs, that still needed a trusted setup phase. The transparency of ZK-STARKs is still upheld due to their characteristic of not having a setup, thus removing the possibility of a single point of failure in the system. ZK-STARKs implement the idea of probabilistic proofs mainly through the FRI algorithm, which helps them come up with short proofs of long computations. Such proofs are verifiable in a time which is sublinear. This makes them extremely effective and scalable, particularly in situations with large data samples like blockchain-based identity management systems. They are also noted to be logically consistent with ZK-SNARKs in terms of quantum resistance. The source of this power is their reliance on randomness and hashing functions for security, unlike elliptic-curve and pairing-based cryptography that they use, which is susceptible to attacks from quantum computers [30]. Hence, due to the fact that no quantum algorithm, to date, has been proven able to efficiently break collision-resistant hash functions, it is safe to say that ZK-STARKs still stand strong against quantum adversaries and can provide a secure basis for the next generations of quantum computers. The deployment of ZK-STARKs casts them as a revolutionizing agent for decentralized apps that demand both privacy and scalability [31], mainly in situations where it is vital that the nature of secret information, like user identities, is not altered while keeping it efficient and secure. Their algorithmic innovation and scalability make ZK-STARKs a critical tool for the future of secure digital identity systems. Table 2 and Figure 1 provides a side-by-side comparison of zk-STARKs and zk-SNARKs, highlighting key differences. The following subsections outline the process for generating and verification a ZK-STARK proof [29].

Table 2. Comparison of ZK-STARKs vs. ZK-SNARKs [32]

Feature	ZK-STARKs	ZK-SNARKs
Trusted Setup	No trusted setup required	Requires trusted setup
Transparency	Fully transparent (no trusted third party)	Relies on a trusted third party for setup
Proof Size	Larger proof size	Smaller proof size
Verification Time	Verifiable in sublinear time	Efficient but slightly longer verification time
Quantum Resistance	Resistant to quantum attacks	Vulnerable to quantum computing attacks
Scalability	Highly scalable, especially with large datasets	Less scalable compared to ZK-STARKs
Mathematical Basis	Based on probabilistic proofs (e.g., FRI – Fast Reed-Solomon IOPP)	Based on elliptic curves and pairings
Efficiency	More efficient for complex computations	Efficient for smaller or simpler computations
Application Areas	Suitable for applications requiring transparency and scalability, like voting systems and identity verification.	Used in Zcash, Filecoin, and Loopring due to small proof size and low storage needs.

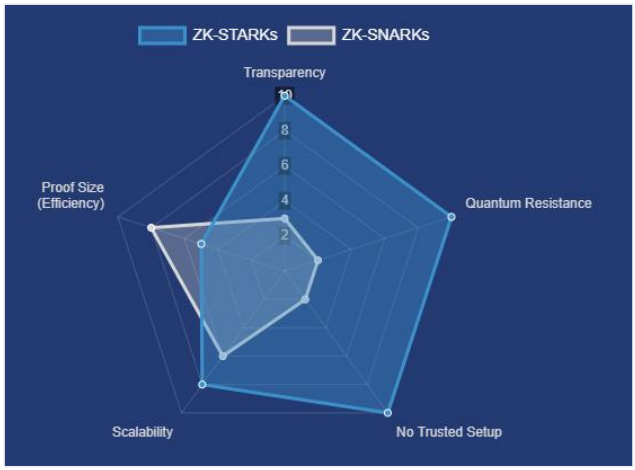

Figure 1. ZK-STARKs vs. ZK-SNARKs

Before diving into the technical mechanics, consider a simple analogy. Imagine you (the Prover) have solved a Sudoku puzzle and want to convince a friend (the Verifier) that your solution is correct without showing them the completed grid.

1. **Commitment:** You write down your entire solution but hide it from your friend. This is the "commitment"; you are now bound to this specific solution and cannot change it.
2. **Verification through Random Sampling:** Instead of looking at the whole grid, your friend randomly asks you to reveal just one specific part—for example, "Show me the third row" or "Show me the top-right 3×3 box."
3. **Proof:** You reveal only the requested row or box. Your friend checks if it contains the digits 1 through 9 exactly once, according to the rules of Sudoku.
4. **Repeat:** You repeat this process several times with different random requests.

After a few successful rounds, your friend becomes mathematically convinced that you have a valid solution for the entire puzzle, even though they have never seen it in its entirety. They have verified your knowledge without you revealing the secret.

ZK-STARKs operate on a similar principle but with mathematical rigor. The "rules of the game" (like the rules of Sudoku) are translated into polynomial equations. This translation process is known as “arithmetization”.

- ZK-STARK Proof Generation

a. Problem Definition and Requirements

The goal of a ZK-STARK proof is to provide a verifiable proof of computation C performed on a dataset D without revealing the dataset itself to ensure privacy. This process allows a prover P (e.g., an institution with confidential data) to prove a claim, such as the absence of a specific data point in D. The proof must reveal nothing about the dataset beyond the confirmed outcome. A specific output α from C (e.g., “no match” for DNA profile p) [29] is shared, while P must ensure that all computation steps align with D without needing trusted intermediaries or revealing sensitive data.

b. Arithmetization-Translating Computation to Algebra

To verify complex computational statements in a verifiable proof, ZK-STARK translates the steps of a computation into an algebraic form, known as arithmetization. This process is broken down as follows:

Algebraic Intermediate Representation (AIR):

AIR uses a set of polynomials $P_i(X,Y)$ to represent each computational step as algebraic constraints. Here, X represents the current state, and Y the subsequent state in the computation. Each transition in the computation (moving from one state to the next) is encoded as:

$$P_1(X,Y) = 0, \quad P_2(X,Y) = 0, \quad \dots, \quad P_s(X,Y) = 0$$

This ensures that every step in the computation aligns with polynomial constraints, forming an algebraic “check” for each operation.

Low-Degree Extension (LDE):

LDE transforms function f, defined over the finite set S, into f^' over a larger set S^', ensuring consistency with the original function over a more extensive range. By extending the polynomial representation of the computation over a larger field or set, ZK-STARK verifies the proximity of this larger dataset to the original. The article emphasizes using the Fast Fourier Transform (FFT) method, particularly the additive FFT for binary fields, to efficiently calculate LDEs.

c. Commitment to Data and Execution Trace

This step ensures that all inputs, intermediate steps, and outputs are committed to in a way that they cannot be altered later. The commitment mechanisms provide a “snapshot” of the data and computation at various points.

Reed-Solomon Encoding:

Data and intermediate steps are encoded using Reed-Solomon codes, creating a mathematical commitment to the computation’s structure and sequence. It secures the trace, allowing verifiers to later check specific parts without needing the entire dataset, making the verification more scalable.

Merkle Tree

A Merkle tree arranges data (encoded states of the computation) in a binary tree, where each node is a hash of its children, and the root hash commits to the entire dataset. This Merkle root acts as a secure, immutable “signature” for all data in the computation, ensuring any alteration in the data can be detected through inconsistent paths back to the root.

$$\text{Commit}(D) = \text{MerkleRoot}(f(D))$$

Authentication Paths

To confirm that specific elements in the computation are correct, each queried element includes an authentication path that shows its alignment with the root commitment.

d. Interactive Oracle Proof of Proximity (IOPP)

Random Queries: Instead of examining all data, the verifier randomly samples certain points in the computation, providing high-confidence verification without full access.

FRI (Fast Reed-Solomon IOP of Proximity): This protocol, discussed in the article, helps identify if the function f approximates a low-degree polynomial g, ensuring the proof is within an acceptable range of accuracy.

$$f \in RS[F,S,\rho] \quad \text{such that} \quad f(x) \approx g(x)$$

e. Proof of Knowledge and Soundness

A ZK-STARK proof must ensure two key properties:

Completeness: If the statement is true (e.g., the DNA is not in the database), the prover can construct a proof that convinces the verifier with high probability.

Soundness: If the statement is false, the prover cannot convince the verifier, except with negligible probability.

The proof must ensure that no information is revealed about the data or the computation beyond the output α. This is

achieved by using randomness in the query responses, ensuring that no sensitive data points are exposed.

- ZK-STARK Proof Verification

a. Merkle Root Verification

The verifier first checks that the Merkle root provided by the prover is consistent with the original committed data. This ensures data integrity, with the root R of the Merkle tree representing the computation trace T satisfying:

$$R = \text{hash}(T)$$

b. Low-Degree Extension (LDE)

The verifier uses a Low-Degree Extension (LDE) to interpolate the polynomial representing the computation trace, ensuring it adheres to a low-degree polynomial.

For a function f defined over a field F with subsets $S \subset S'$, the LDE f' is represented by:

$$f'(x) = \sum_{i=0}^{d} a_i \, x^i$$

The complexity of the LDE is optimized using the Fast Fourier Transform (FFT), reducing operations to [29]:

$$3 \cdot |S'| \log |S'|$$

c. Reed-Solomon Interactive Oracle Proof of Proximity (FRI Protocol)

To confirm that the prover's polynomial $f(x)$ maintains a low degree, the FRI protocol performs proximity testing by sampling random values $x_1, x_2, \ldots, x_k$.

The verifier ensures that $f(x)$ approximates a polynomial of degree $< \rho|S|$, where ρ is the rate parameter:

$$f(x) \approx g(x)$$

d. Random Sampling with Merkle Path Verification

For each random point x_i queried, the verifier uses the Merkle paths from leaf to root to confirm the consistency with the committed trace.

Path verification follows:

$$\text{hash}(x_{i-1}, x_i) = x_{\text{parent}}$$

Each path must hash to the correct parent node, ensuring consistency with the initial commitments.

e. Final Consistency Check

After passing all polynomial and Merkle checks, the verifier performs a final consistency test to ensure all constraints for each queried polynomial point align with the initial commitments.

This final check confirms the integrity and validity of the entire computation trace, completing the ZK-STARK verification process.

The following ratios determine ZK-STARK's efficiency in verification time and communication size:

$$\rho_{\text{time}} = \frac{T_V}{T_C} \quad \text{and} \quad \rho_{\text{size}} = \frac{CC}{|D|}$$

3.5 InterPlanetary File System (IPFS)

The InterPlanetary File System (IPFS) is the technology that is being extensively applied in different scopes of work to get the most out of its decentralized, peer-to-peer storage architecture [33]. One of the most outstanding cases in the world of IdM is the utilization of Filecoin, A decentralized storage network that is built upon the IPFS protocol for accessing the content distribution platforms. In the Filecoin network, users can pool their resources for securing and retrieving files in a decentralized fashion, which helps in reducing the need for central data servers. This strategy is more efficient and sustainable in the manner it protects against data outages and censorship because the files are distributed among different nodes, instead of being hosted on a single server.

The other significant example is a DTube, completely decentralized video hosting platform for creating and distributing videos, which is operating like Youtube but using IPFS for storing the video content. By making the video files distributed among nodes, DTube assures the content is not controlled or censored by a single entity, thus the users get more control and privacy. So, the IPFS is also used in the decentralized version of web hosting, where the websites exist as distributed across a network of nodes, which guarantees that the sites are always up even if some of the nodes go down. These use cases vividly showcase IPFS's ability to make the life of web applications more resilient, privacy-preserving, and decentralized.

The methodology that IPFS employs is to split the files up into several pieces, which are then each given a unique cryptographic hash called a content identifier (CID) [34]. These chunks are distributed across different nodes in the network, and when a user requests a file, the network reassembles it from the distributed pieces using the content identifiers. Since the system is content-addressed, users retrieve data based on the hash of the content itself, not its location on a server. This ensures that files are immutable; any change in the file would produce a different hash, effectively creating a new version of the file. This process ensures data integrity, as any tampering is immediately detectable through hash comparison. Additionally, IPFS uses a distributed hash table (DHT) [35] to locate which nodes store specific chunks of data, ensuring efficient and scalable retrieval.

4. ARCHITECTURE OVERVIEW

The decentralized identity management system enables users to manage their own digital identities securely and privately using a blockchain infrastructure. Our architecture incorporates zk-STARKs to ensure that users can verify their identity attributes without revealing sensitive information, while the use of DIDs and IPFS ensures that users maintain full control over their identities without the need for centralized authorities. In the following sections we will define our architecture key components, and how the integration of zk-STARKs and IPFS enhance the privacy, availability and security of user's identity. Sign up and verification flow are also described on the next sections.

4.1 Key components

The main components of our proposed architecture are:

User: The person who interacts with the decentralized application (Dapp) via their wallet (e.g., MetaMask). The User provide necessary personal data (name, location, and birthday) through a sign-up form and also authorize transactions by

connecting their wallet.

Wallet: A cryptographic wallet that holds the user's private key and enables them to sign transactions on the Ethereum blockchain. It connects to the Dapp, allowing the user to register and approve smart contract interactions. It also provides the wallet address (GetWallet () function) used for generating Decentralized Identifiers (DID).

BLOCKID Dapp: The frontend application interface that allows users to interact with the system. It acts as the client-side interface where users connect their wallet and fill out the sign-up form with personal details (name, location, birthday). It communicates with the smart contract backend to execute operations such as DID generation, data submission, and IPFS storage.

Smart Contract: A decentralized program running on the Ethereum blockchain responsible for the logic of the Dapp. It's used for:

- *Generate DID:* Creates a unique Decentralized Identifier (DID) for each user based on their wallet address. This ensures that each user has a verifiable identity tied to their wallet.
- *Push DID:* Once generated, the DID is pushed onto the blockchain using the wallet address as a reference.
- *Submit Data:* The personal details (name, location, birthday) are encrypted using zk-STARKS and submitted to the contract.
- *Store Hash:* After the data is encrypted, it is stored on IPFS, and the resulting IPFS hash is saved in the smart contract along with the user's DID.

IPFS (InterPlanetary File System): A decentralized storage system that holds the encrypted personal data. After the user data (name, location, birthday) is encrypted using zk-STARKS, it is stored on IPFS in a distributed manner. An IPFS hash is generated, which uniquely identifies the stored data [33].

4.2 Sign-up workflow

As shown in Figure 2, the sign-up process is described as below:

1. The user initiates the process by accessing the BLOCKID.
2. The user connects their Ethereum wallet (e.g., MetaMask, TrustWallet) to the DApp to authenticate. The wallet's Ethereum address is retrieved and stored locally in the DApp as the user's on-chain identity.
3. The DID is generated by taking the user's Ethereum address using the W3C DID specification and applying a prefix (did:blockid:) that aligns with Ethereum DID methods.
4. The DApp sends the generated DID and the Ethereum wallet address to a Smart Contract deployed on Ethereum. The contract maintains a mapping of Ethereum addresses to their corresponding DIDs.
5. User fills out the sign-up form (Name, Location, Birthday) and submits it.

Example:

```
{
"Name": "Mohammed",
"location": "Morocco",
"birthday": "1964-12-11"
}
```

6. The user's name, location, and birthday are prepared for encryption. The user data is converted into a format suitable for zk-STARK encryption. The data is transformed into polynomials over a finite field. This process involves expressing the user's data as a set of equations that can be later used for zero-knowledge proofs. For example, the name "Mohammed" is hashed and represented as:

$$P_{\text{name}}(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$

7. The DApp takes the polynomial representations and generates a succinct proof that the data satisfies a particular set.
8. The DApp immediately validates the generated zk-STARK proof to ensure its correctness and integrity:

- If the proof is valid: The workflow continues to the next step.
- If the proof is invalid: The process is halted. An error is communicated to the user, preventing the system from storing invalid or corrupted data on IPFS. The user may be prompted to resubmit their information.

9. The user data (name, location, birthday), along with the zk-STARK proof, is packaged and then encrypted for extra security before storing on IPFS, ensuring the data cannot be accessed without the encryption key.
10. The encrypted data is uploaded to IPFS, and a Content Identifier (CID) is generated, which acts as the reference to the stored data.
11. The IPFS hash and zk-STARK proof are now linked to the user's DID and pushed to the blockchain. The smart contract can maintain a mapping of DID to IPFS hashes.

4.3 Verification workflow

Figure 3 illustrates the proposed system's verification workflow:

1. The user initiates the process by requesting an access to a service provider (verifier). The verifier could be an external service that requires identity verification (e.g., a banking service, government portal, etc.).
2. The user authenticates with the verifier by connecting their wallet used on BLOCKID App (e.g., MetaMask).
3. The verifier retrieves the wallet address of the user to establish their identity.
4. The verifier queries the Smart Contract on the Ethereum blockchain to retrieve the DID and the linked IPFS hash associated with the user's Ethereum wallet.
5. The Smart Contract returns the IPFS hash to the external service.
6. The verifier sends a request to IPFS to retrieve the encrypted identity data linked to that hash.
7. The IPFS sends back the encrypted identity data and the ZK-STARK proof associated with the user's DID.

Example:

```
{
    "proof": "zk_stark_proof_data",
    "encrypted_data": {
        "name": Encrypted (Mohammed),
        "location": Encrypted (Morocco),
        "birthday": Encrypted (1964-12-11)
    }
}
```

8. The verifier verifies the ZK-STARK proof to ensure the integrity and authenticity of the encrypted data without actually revealing the sensitive information. The proof is validated off-chain to reduce Ethereum gas fees.
9. If ZK-STARK proof is valid, the verifier can confidently confirm the user's identity and grant access to the requested service. If K-STARK proof fails, the user is denied access as the verification process did not confirm their identity.

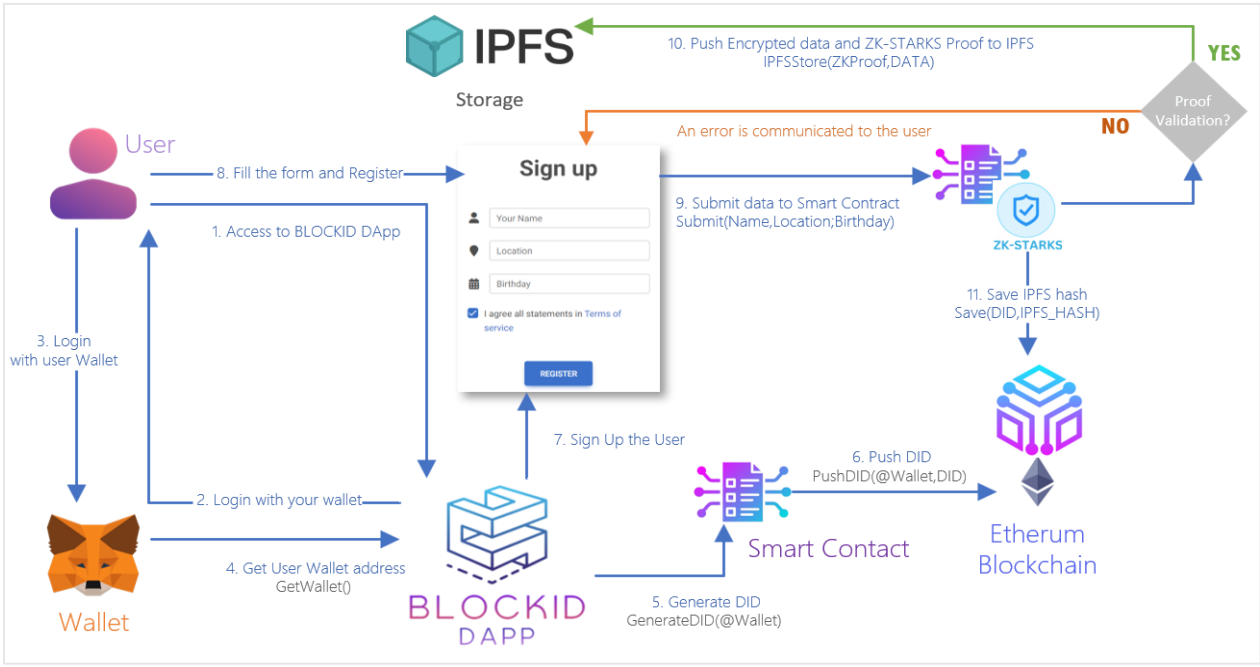

Figure 2. Proposed system sign-up workflow

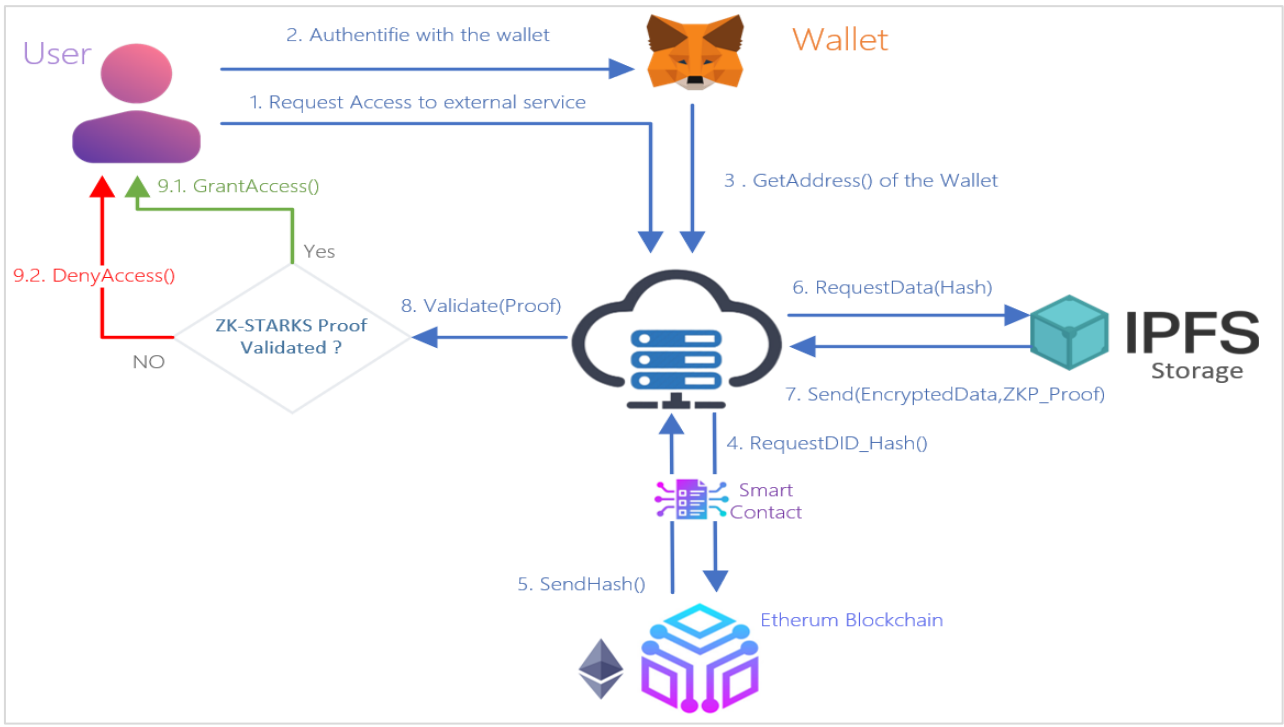

Figure 3. Proposed system verification workflow

5. IMPLEMENTETION

5.1 Overview of development setup

The primary development was performed on a machine equipped with a 13th Gen i7-1355U processor, 16 GB of installed RAM, and running the Windows 11 Pro operating system. This configuration represents a standard development environment, indicating that the system's construction does not necessitate specialized or prohibitively expensive hardware. This aspect is pertinent as it suggests broader accessibility for future research, development, and potential adoption of similar architectures. The core technology stack selected for this project comprises (Table 3) Ganache for local blockchain simulation [36], IPFS Desktop for managing decentralized file storage [37], React.js for crafting the decentralized application front-end, and the Winterfell library [38], for the generation and verification of ZK-STARKs.

5.2 Zkstark library

In the critical selection of a ZK-STARK library, a rigorous evaluation of available options was undertaken, primarily focusing on libSTARK and Winterfell. libSTARK, a C++

implementation [39] originating from the authors of the foundational STARK academic paper, was initially considered for its direct lineage. However, preliminary assessments raised concerns regarding its perceived reliance on a specialized circuit design notation, which could potentially impede seamless integration with the project's established programming paradigm. Furthermore, libSTARK's self-characterization as "academic grade" and its explicit disclaimer of likely containing "multiple serious security flaws" rendered it less optimal for the development of a robust prototype.

Table 3. Tools used for prototype implementation

Tool	Version	Role in Project	Benefit for This Project
Ganache [36]	2.7.1	Local Ethereum environment for smart contract deployment, testing, and DApp interaction.	Enables rapid, cost-free iterative development and testing in a controlled, deterministic environment.
IPFS Desktop [37]	0.39.0	Managing local IPFS node for storing and retrieving encrypted user data off-chain.	Simplifies IPFS usage via GUI, facilitating decentralized storage management integral to the system's privacy model.
React.js		Building the user interface for registration, data input, and wallet interaction.	Facilitates modular UI development with a rich ecosystem for web3 and IPFS integration.
Winterfell [38]	0.8.1	Generating and verifying ZK-STARK proofs for privacy-preserving identity attribute validation.	Chosen for its Rust implementation, perceived good documentation, active development, and STARK capabilities.

Consequently, the Winterfell library [38], a Rust-based crate, emerged as the preferred alternative. This decision was predicated on several compelling factors: Winterfell presented as a well-implemented solution, distinguished by provided documentation, and was in active development. It is engineered to offer a relatively simple interface for describing general computations and exhibits advantageous genericity over finite fields and hash functions, thereby affording greater implementation flexibility. The choice of Rust, a language increasingly recognized for its performance and memory safety attributes in cryptographic contexts, further solidified Winterfell's suitability [40]. Ease of integration offered by Winterfell provided a more pragmatic and convincing pathway for the successful incorporation of advanced ZK-STARK cryptographic proofs into the proposed identity management framework.

5.3 Smart contracts

The on-chain logic of the decentralized identity management system is articulated through smart contracts, which were meticulously authored in the Solidity programming language, specifically targeting version 0.8.0 or compatible iterations thereof. For the systematic deployment and instantiation of these smart contracts onto the Ganache local test blockchain, the Truffle development suite was employed. Truffle's migration scripts facilitated the staging and execution of deployment tasks, ensuring the compiled Solidity bytecode was correctly migrated to the Ganache network, typically by configuring Ganache as a designated network within the truffle-config.js file [41, 42] and subsequently invoking the truffle migrate --network command.

An exemplary smart contract, DIDManager, serves as the cornerstone for managing Decentralized Identifiers (DIDs) within the proposed architecture. This contract is responsible for the lifecycle management of DIDs, including their creation, revocation, and retrieval. It maintains a historical record of DIDs associated with user addresses, ensuring that each user can possess only one active DID at any given time. The contract defines a DIDRecord structure to store pertinent information such as the DID string, creation and revocation timestamps, and an activity status flag. Key functionalities include createDID (Figure 4) for registering a new DID, revokeDID for deactivating an existing DID, and getter functions like getActiveDID and getDIDHistory to query the status and historical data of DIDs. Events such as DIDCreated and DIDRevoked are emitted to log significant state changes, enhancing transparency and auditability. Modifiers are employed to enforce preconditions, such as ensuring a user has no active DID before creating a new one, or possesses an active DID before attempting revocation.

DIDManager Code

```
pragma solidity ^0.8.0;

contract DIDManager {
    struct DIDRecord {
        string did;
        uint256 creationDate;
        uint256 revocationDate;
        bool active;
    }
    //Mapping from address to their DID history
    mapping(address=>DIDRecord[]) private didHistory;

    //Events
    event DIDCreated (address indexed owner, string did,
uint256 timestamp);
    event DIDRevoked (address indexed owner, string did,
uint256 timestamp);
    // Modifiers
    modifier hasNoActiveDID () {
        bool hasActive=false;
        if (didHistory[msg.sender]. length>0) {
            hasActive=
didHistory[msg.sender][didHistory[msg.sender].          length-1].
active;
        }
        require (!hasActive, "An active DID already exists");
        _;
    }
    modifier hasActiveDID () {
        bool hasActive=false;
        if (didHistory[msg.sender]. length>0) {
            hasActive                                          =
didHistory[msg.sender][didHistory[msg.sender].          length-1].
active;
        }
        require (hasActive, "No active DID found");
        _;
    }

    // Create a new DID
    function createDID (string memory did) external
```

```
hasNoActiveDID {
        DIDRecord memory newRecord = DIDRecord ({
           did: did,
           creationDate: block.timestamp,
           revocationDate: 0,
           active: true
        });

        didHistory[msg.sender]. push (newRecord);
        emit DIDCreated (msg.sender, did, block.timestamp);
    }

    // Revoke the current active DID

    function revokeDID () external hasActiveDID {
        uint256 lastIndex=didHistory[msg.sender]. length-1;
        didHistory[msg.sender][lastIndex]. active=false;
        didHistory[msg.sender][lastIndex].      revocationDate=
block. timestamp;

        emit DIDRevoked (
           msg.sender,
           didHistory[msg.sender][lastIndex]. did,
           block.timestamp
        );

    }
    //Get the currently active DID
    function getActiveDID (address user) external view returns
(DIDRecord memory) {
        require(didHistory[user]. length>0, "No DID history
found");

        DIDRecord memory lastRecord =
didHistory[user][didHistory[user]. length-1];
        require (lastRecord.active, "No active DID found");

        return lastRecord;
    }
    //Get the complete DID history for an address
    function getDIDHistory(address user) external view returns
(DIDRecord[] memory) {
        return didHistory[user];
    }
}
```

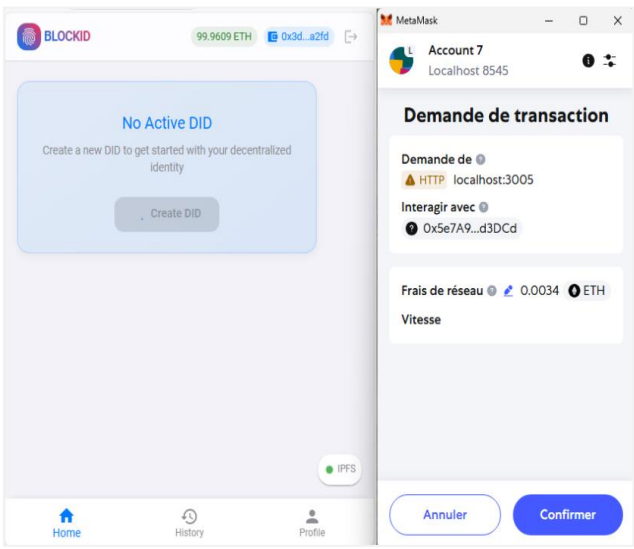

Figure 4. DIDManager contract call createDID

Table 4 presents the gas consumption of key functions in the DIDDataManager contract. Both createDID and add Data Hash ToDID used their full gas limits (151,047 and 117,148), indicating predictable execution. The revokeDID function used 59,932 out of 64,732 gas, completing with unused capacity suggesting conditional or less intensive operations.

Table 4. Gas usage of DIDDatamanager smart contract functions executed on ethereum

Function	Gas Limit	Gas Used	Block
createDID	151047	151047	226
addDataHashToDID	117148	117148	228
revokeDID	64732	59932	224

5.4 IPFS integration

A pivotal component of the system's privacy [42] enhancing architecture involves the secure off-chain management of user data, leveraging the IPFS for decentralized storage. To uphold principles of user sovereignty and fortify data security, a user-centric cryptographic protocol is instituted. As depicted in the user interface for profile updates (Figure 5), when an individual submits or modifies their personal attributes via the DApp, a cryptographic signature is solicited. The user's wallet prompts them to sign a specific message, explicitly stating, "Sign this message to derive your encryption key". This signature, uniquely generated via the user's private key, serves as the basis for deriving a symmetric encryption key. Subsequently, the user's personal data is encrypted client-side within the DApp environment using this derived key, employing robust cryptographic standards such as AES-256, a detail corroborated by the system's output metadata (Figure 6).

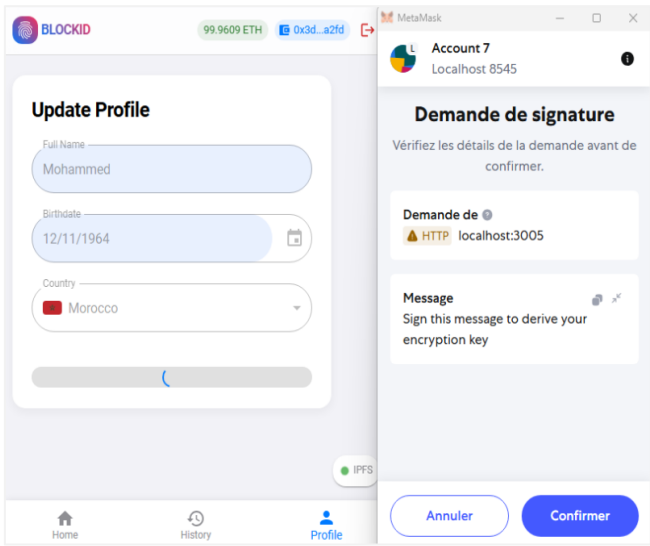

Figure 5. Signature prompt

Following client-side encryption, the resultant encrypted data package is committed to the IPFS network [43]. Upon successful ingestion, IPFS furnishes a unique Content Identifier (CID), exemplified by values such as QmNNGd1ouzCB6d5SnvgeDMuqvXE6RiL5N4K9Umw3PhgK9. This CID functions as an immutable and content-addressed pointer to the encrypted data payload residing on the distributed IPFS network. The final step in this workflow involves the on-chain registration of this CID, associating it with the user's DID within the smart contract. This architectural design ensures that sensitive user information remains confidential and under the user's exclusive control,

decipherable only with the key derived from their unique signature, while the integrity and verifiability of the data's existence and linkage are maintained by the blockchain and IPFS.

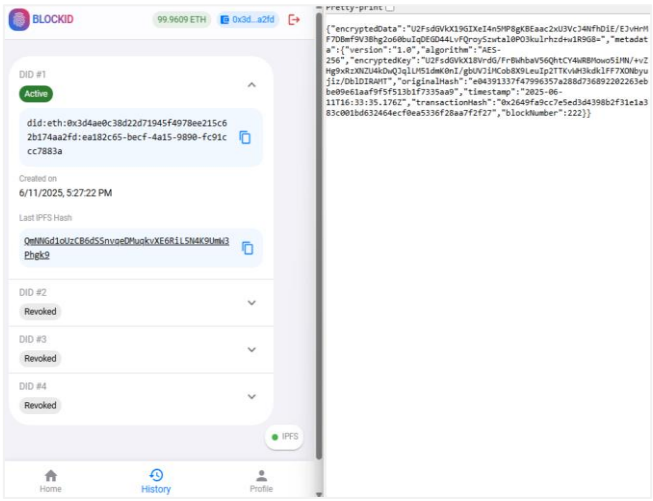

Figure 6. IPFS CID hash and content

Table 5 summarizes the IPFS storage metrics for a typical encrypted identity record. The file size was 551 bytes, and both the CID generation and upload processes were completed rapidly within 5ms and 8ms respectively using a local IPFS node. The content was encoded using the dag-pb (UnixFS) codec and hashed with the sha2-256 multihash function, resulting in a 256-bit (32-byte) digest. These results demonstrate the suitability of IPFS for fast and lightweight off-chain storage in decentralized identity systems [44].

Table 5. IPFS storage metric

Metric	Value
Size	551 bytes
Generation time	5ms
Upload time	8ms
Codec	dag-pb (UnixFS)
Multihash Function	sha2-256
Digest Size	256 bits (32 bytes)

5.5 Winterfell library

In this prototype, we used Winterfell for age verification, a process that leverages zero-knowledge STARKs to confirm an individual's is adult without revealing their birthdate. The core logic, which is a simplified calculation like current_year-birth_year≥18, is translated into an Algebraic Intermediate Representation (AIR). This arithmetization is a fundamental step in zk-STARK systems, ensuring that a computational statement is true through a set of polynomial constraints. The user's birthdate acts as a private input for the prover, while public inputs include the current date and the required age threshold, in this case, 18. The Winterfell library's documentation outlines the general methodology for defining an AIR and constructing the necessary execution trace.

The fundamental relationship we need to prove is:

current-birthdate≥EIGHTEEN_YEARS_IN_SECONDS

where, *current* is a public input, known to both the Prover and Verifier. *birthdate* is a private input (the witness), known only to the Prover. EIGHTEEN_YEARS_IN_SECONDS is a public constant, defined as 568,036,800.

STARKs are most efficient at proving polynomial equalities, not inequalities. Therefore, we transform the inequality into an equality and a range check. We introduce a new variable, *remainder*, such that:

current-birthdate=EIGHTEEN_YEARS_IN_SECONDS+*remainder.*

By rearranging, we get:

remainder=*current*-EIGHTEEN_YEARS_IN_SECONDS-*birthdate*

If we can prove that *remainder* ≥0, we have successfully proven the original inequality. Proving *remainder* ≥0 is accomplished via a range check, where we prove that remainder is a value that can be expressed within a certain number of bits (e.g., 63 bits), which mathematically guarantees it is a non-negative number within the finite field.

Table 6. Impact of blowup factor on STARK proof generation, size, and verification time

Blowup Factor	Proof Generated (ms)	Proof Size (KB)	Proof Verified (ms)
8	27	15.7	0.7
16	30	18.3	1.0
32	32	21.0	1.2
64	33	23.4	1.4
128	37	26.5	1.5

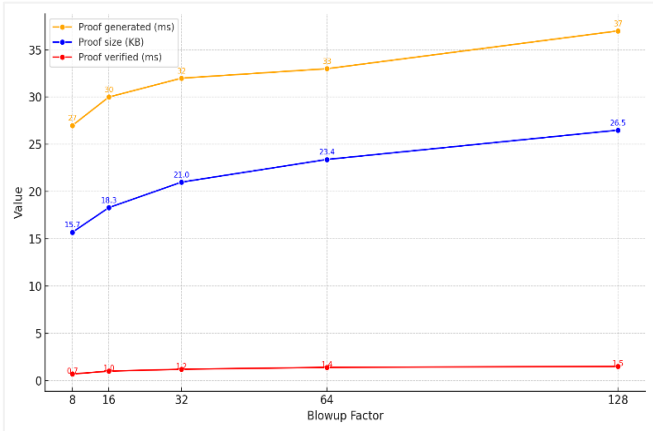

Figure 7. Performance metrics vs. blowup factor in ZK-STARK proofs

To measure the system's real-world performance, we conducted benchmarks on a standard laptop with 16GB RAM. Our analysis focuses on a minimal computation, represented by a trace length of just 7. For such a small trace, the proof's integrity does not stem from the complexity of the computation itself, but rather from its robust cryptographic security parameters. Table 6 illustrates how performance scales with the blowup factor, a critical parameter that directly governs the trade-off between prover workload and the security level. By expanding the trace's evaluation domain, a larger blowup factor makes it significantly harder for a dishonest prover to forge a proof. To isolate its specific impact, the blowup factor was varied while other crucial security parameters, such as the number of queries (28), were kept

constant.

As shown in Figure 7, increasing the blowup factor leads to a gradual rise in both proof size and generation time, which is expected due to the higher computational demands required for stronger soundness guarantees. However, the proof verification time remains relatively stable ranging only from 0.7ms to 1.5ms despite the growing size and complexity of the proofs.

6. DISCUSSION

Table 7 provides a comprehensive comparison between the proposed Blockchain, zk-STARK and IPFS based identity management architecture and several related works. Each works is evaluated based on its technical implications in privacy preservation, scalability, quantum resistance, and data integrity. By exploring them, we underline how our framework stands out, providing a secure, decentralized, and future-proof solution. The proposed architecture represents a prototype solution that addresses many of the limitations identified in existing identity management systems. By combining Blockchain, zk-STARKs and IPFS, this architecture enhances privacy, scalability, and decentralization, offering distinct advantages across key indicators.

Table 7. Comparison of proposed identity management system (Blockchain, zk-STARK and IPFS) vs. related works

Feature	Proposed System	[14]	[13]	[18]	[17]	[16]
Privacy Preservation	H	M	L	H	M	M
Quantum Resistance	H	L	-	M	M	L
Setup Transparency	H	L	L	H	M	M
On-Chain Data Minimization	H	M	M	M	H	M
Verification Speed	H	M	M	H	M	L
Scalability	H	L	M	M	M	L
User Control & Decentralization	H	M	L	H	H	M
Security	H	M	L	H	H	M
Interoperability	H	M	L	M	M	M
Cost Efficiency	H	L	M	H	M	M
Data Immutability & Integrity	H	M	L	H	H	M

H: High; M: Moderate; L: Low

In evaluating privacy preservation, our proposed system demonstrates a distinct advantage due to the integration of zk-STARKs, which ensures high levels of zero-knowledge privacy for identity verification. This level of privacy preservation outperforms many existing systems that still rely on traditional cryptographic methods, often requiring trusted setups that may not fully protect user data. Some related works have demonstrated moderate privacy capabilities by employing zk-SNARKs or other privacy-enhancing techniques. However, these approaches still risk exposing sensitive data to trusted third parties for validation [11, 12].

Regarding quantum resistance, our system is built using zk-STARKs, which are inherently resistant to quantum attacks. This distinguishes it from other solutions based on zk-SNARKs, as highlighted in studies such as [17]. While zk-SNARKs remain effective against current cryptographic challenges, their reliance on elliptic curve cryptography makes them potentially vulnerable to future quantum computing threats. In contrast, our system's quantum resistance is derived from its reliance on collision-resistant hash functions rather than elliptic curves, positioning it as a more future-proof solution for identity management [18, 28].

Setup transparency also highlights a critical distinction. Our proposed prototype requires no trusted setup. This transparent setup model provided by using zk-STARKs minimizes dependency on centralized trust models and offers a reliable framework that enhances user trust. In contrast, several related identity management systems still depend on trusted setups, which can introduce potential vulnerabilities like single point of failure [16]. By eliminating the need for such setups, our approach aligns with the goal of full transparency in decentralized identity verification.

On-chain data minimization is a significant advantage of our approach. In our prototype, only the IPFS hash and user DID are saved on the blockchain, reducing the amount of sensitive data on-chain and alleviating potential blockchain bloat issues. This strategy contrasts with some related solutions [11, 16] that still retain user data or verifiable attributes on-chain, which can lead to exposure risks. Our method provides a practical balance by securing sensitive data off-chain, stored on IPFS while maintaining verification integrity on the blockchain.

In terms of scalability, the proposed architecture demonstrates the capacity to efficiently support larger datasets, primarily due to the stable and rapid verification performance afforded by zk-STARKs. As evidenced by the prototype evaluation results (Table 7), increases in proof size—resulting from higher blowup factors—lead to only marginal increases in verification time, which remains consistently low, ranging from 0.7ms to 1.5ms. This stability indicates that the verification process is not significantly impacted by the complexity or size of the proof, thereby ensuring sustained performance under high-load conditions. Additionally, the integration of IPFS for off-chain storage enables the system to manage large encrypted identity data without overburdening the blockchain. This architectural choice contrasts with prior systems employing zk-SNARKs [13, 11], which frequently encounter scalability limitations due to higher computational overhead and slower verification times as proof complexity increases.

Unlike the centralized systems critiqued by Anusuya [16], which restrict user autonomy, our model is architected to empower the individual. Through the integration of decentralized identifiers (DIDs) and off-chain IPFS storage, we shift the locus of control away from central authorities, thereby enabling users to manage their own data. This design philosophy aligns our work directly with the principles of decentralized identity management.

In terms of security evaluation, our architecture offers strong cryptographic assurances through zk-STARKs, which not only ensure that the data is intact but also do not require a trusted setup. This is different from certain current systems that are secure, however, they still have vulnerabilities due to their dependence on trusted setups and are at risk of quantum attacks. Our method plays a part in minimizing these hazards by spreading out the identity management operations and employing cryptography that is resistant to quantum [16].

Because our proposed system employs open standards such as DIDs and zk-STARKs, which are interoperable with several decentralized protocols, the interoperability with other decentralized applications is improved to a great extent. However, the BiDaas system which is also decentralized, has some shortcomings in the integration of different ecosystems due to it still using traditional verification methods. Our

system's compatibility with decentralized protocols makes it versatile and adaptable, ideal for cross-platform identity verification use cases.

Cost efficiency represents a significant advantage of the proposed system. By minimizing on-chain storage through the use of IPFS, where only the CID and DID are stored on-chain, the architecture substantially reduces gas consumption during identity verification operations. This off-chain strategy avoids the high costs typically incurred by systems that store user attributes or verifiable credentials directly on the blockchain. In the prototype implementation, encrypted identity records averaged just 551 bytes and were uploaded to IPFS with a total processing time of under 13 milliseconds. Furthermore, the lightweight nature of zk-STARK verification remaining within the 0.7ms to 1.5ms range regardless of proof size translates to lower computational demands during validation. Combined, these characteristics reduce both on-chain transaction fees and the processing overhead required for proof verification. In contrast, systems that rely on zk-SNARKs and on-chain data retention often experience elevated costs, particularly in large-scale deployments.

Finally, data immutability and integrity are strengthened in our system by leveraging zk-STARKs for cryptographic proofs and IPFS for secure storage. This dual approach ensures that data remains unaltered once uploaded, a critical feature for identity management systems where data tampering could have severe consequences. While Anusuya [16] discusses data immutability in centralized systems, our decentralized model provides an additional layer of reducing risks associated with single points of failure.

7. CONCLUSION

In conclusion, the proposed design that drew strengths from Blockchain, zk-STARKs, and IPFS presents a highly innovative and scalable approach that logically solves the age-old problems in the identity management field. As illustrated in the comparative analysis in Table 7, our framework offers significant advantages over existing systems. Explicitly contrasting with the related works, our system achieves a high degree of privacy preservation by using zk-STARKs, which, unlike the zk-SNARK-based approaches in other studies, do not require trusted setups that can expose user data. A primary differentiator is our framework's high quantum resistance, a critical feature for future-proofing digital identity. This stands in direct opposition to several referenced systems that remain vulnerable to quantum computing attacks due to their reliance on traditional cryptographic methods. Furthermore, our model ensures high setup transparency and decentralization, eliminating the risks associated with a single point of failure inherent in systems requiring a trusted setup. The performance of our architecture is built upon two pillars: radical on-chain data minimization via IPFS and highly efficient computation using zk-STARKs. First, by storing only a hash and a DID on-chain, we mitigate the cost and privacy risks of blockchain bloat. The efficiency of this off-chain approach was validated in our prototype, which processed and stored encrypted identity records (avg. 551 bytes) in under 13ms. Second, zk-STARKs handle complex computations with minimal overhead, ensuring that verification speeds do not degrade at scale. Our benchmarks demonstrate this clearly, with stable verification times (0.7ms-1.5ms) despite increasing proof sizes (15.7KB to 26.5KB). This dual architecture provides a cohesive solution to the security, privacy, and scalability challenges that pervade existing systems.

REFERENCES

[1] Maidine, K., El-Yahyaoui, A. (2023). Cloud identity management mechanisms and issues. In 2023 IEEE 6th International Conference on Cloud Computing and Artificial Intelligence: Technologies and Applications (CloudTech). Marrakech, Morocco, pp. 1-9. https://doi.org/10.1109/CloudTech58737.2023.10366178

[2] Maidine, K., El-Yahyaoui, A. (2024). Key mechanisms and emerging issues in cloud identity systems. In International Conference of Cloud Computing Technologies and Applications. Springer, Cham, pp. 64-88. https://doi.org/10.1007/978-3-031-78698-3_5

[3] Over 184m passwords from Apple, Google, Facebook, Microsoft exposed in massive leak. (2025). Hindustan Times. https://www.hindustantimes.com/technology/over-184m-passwords-from-apple-google-facebook-microsoft-exposed-in-massive-leak-101748580879587.html, accessed on Jun. 12, 2025.

[4] Over 184m passwords from Apple, Google, Facebook, Microsoft exposed in massive leak. (2025). Hindustan Times. https://www.hindustantimes.com/technology/over-184m-passwords-from-apple-google-facebook-microsoft-exposed-in-massive-leak-101748580879587.html, accessed on Jun. 12, 2025.

[5] Dhaliwal, J. (2025) Adidas data breach: What consumers need to know and how to protect yourself, McAfee Blog. https://www.mcafee.com/blogs/internet-security/adidas-data-breach-what-consumers-need-to-know-and-how-to-protect-yourself/, accessed on Jun. 12, 2025.

[6] Select Medical Holdings Data Breach Lawsuit Investigation. (2025). ClassAction. org. https://www.classaction.org/data-breach-lawsuits/select-medical-holdings-june-2025, accessed on Jun. 12, 2025.

[7] Cluley, G. TeleMessage, the signal clone used by US government officials, suffers hack, Hot for Security. https://www.bitdefender.com/en-us/blog/hotforsecurity/telemessage-signal-clone-us-government-hack, accessed on Jun. 12, 2025.

[8] PLLC, S.B. (2025). LexisNexis Risk solutions data breach investigation-Strauss Borrelli PLLC, Strauss Borrelli PLLC-. https://straussborrelli.com/2025/05/28/lexisnexis-risk-solutions-data-breach-investigation/, accessed on Jun. 12, 2025.

[9] Thorve, A., Shirole, M., Jain, P., Santhumayor, C., Sarode, S. (2022). Decentralized identity management using blockchain. 2022 4th International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), Greater Noida, India, pp. 1985-1991. https://doi.org/10.1109/ICAC3N56670.2022.10074477

[10] El-Yahyoui, A., EC-Chrif El Kettani, M.D. (2017). Fully homomorphic encryption: Searching over encrypted cloud data. In 2nd international Conference on Big Data, Cloud and Applications, Tetouan, Morocco. https://doi.org/10.1145/3090354.3090364

[11] Ma, S., Zhang, X. (2024). Integrating blockchain and ZK-ROLLUP for efficient healthcare data privacy protection system via IPFS. Scientific Reports, 14(1): 11746. https://doi.org/10.1038/s41598-024-62292-9
[12] Seddik, S., Routaib, H., Elmounadi, A., Haddadi, A.E. (2024). Enhancing African market predictions: Integrating quantum computing with Echo State Networks. Scientific African, 25: e02299. https://doi.org/10.1016/j.sciaf.2024.e02299
[13] Lee, J.H. (2017). BIDaaS: Blockchain based ID as a service. IEEE Access, 6: 2274-2278. https://doi.org/10.1109/ACCESS.2017.2782733
[14] Yang, X., Li, W. (2020). A zero-knowledge-proof-based digital identity management scheme in blockchain. Computers & Security, 99: 102050. https://doi.org/10.1016/j.cose.2020.102050
[15] Barros, M.D.V., Schardong, F., Custódio, R.F. (2022). Leveraging self-sovereign identity, blockchain, and zero-knowledge proof to build a privacy-preserving vaccination pass. arXiv Preprint arXiv: 2202.09207. https://doi.org/10.48550/arXiv.2202.09207
[16] Anusuya, R., Karthika Renuka, D., Ghanasiyaa, S., Harshini, K., Mounika, K., Naveena, K.S. (2021). Privacy-preserving blockchain-based ehr using zk-snarks. In International Conference on Computational Intelligence, Cyber Security, and Computational Models. Cham: Springer International Publishing. Springer, Cham, pp. 109-123. https://doi.org/10.1007/978-3-031-15556-7_8
[17] Lohar, S. (2024). Decentralization of identity using ethereum and IPFS. Communications on Applied Nonlinear Analysis, 31(4s): 378-391. https://doi.org/10.52783/cana.v31.917
[18] Zhu, X., He, D., Bao, Z., Luo, M., Peng, C. (2023). An efficient decentralized identity management system based on range proof for social networks. IEEE Open Journal of The Computer Society, 4: 84-96. https://doi.org/10.1109/OJCS.2023.3258188
[19] Panait, A.E., Olimid, R.F. (2020). On using zk-SNARKs and zk-STARKs in blockchain-based identity management. In International Conference on Information Technology and Communications Security. Cham: Springer International Publishing, pp. 130-145. https://doi.org/10.1007/978-3-030-69255-1_9
[20] Wright, C.S. (2008). Bitcoin: A peer-to-peer electronic cash system. SSRN Electronic Journal, 3440802: 10-2139. https://doi.org/10.2139/ssrn.3440802.
[21] Nakamoto, S. (2009). Bitcoin: A peer-to-peer electronic cash system. Bitcoin. org. Disponible En. https://bitcoin.org/en/bitcoin-paper.
[22] Garg, R. (2023). Blockchain ecosystem. In Blockchain for Real World Applications, pp. 23-42. https://doi.org/10.1002/9781119903765.ch3
[23] Ramachandran, M. (2025). Introduction to blockchain concepts. In Blockchain technologies, pp. 3-33. https://doi.org/10.1007/978-981-96-4360-8_1
[24] Abdelgalil, L., Mejri, M. (2023). HealthBlock: A framework for a collaborative sharing of electronic health records based on blockchain. Future Internet, 15: 87. https://doi.org/10.3390/fi15030087
[25] Atzei, N., Bartoletti, M., Cimoli, T. (2017). A survey of attacks on Ethereum Smart Contracts (SoK). In Principles of Security and Trust. POST 2017. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-54455-6_8
[26] Ji, S., Wu, J., Qiu, J., Dong, J. (2023). Effuzz: Efficient fuzzing by directed search for smart contracts. Information and Software Technology, 159: 107213. https://doi.org/10.1016/j.infsof.2023.107213
[27] Zhou, H., Milani Fard, A., Makanju, A. (2022). The state of ethereum smart contracts security: Vulnerabilities, countermeasures, and tool support. Journal of Cybersecurity and Privacy, 2(2): 358-378. https://doi.org/10.3390/jcp2020019
[28] Dhar, S., Khare, A., Dwivedi, A.D., Singh, R. (2024). Securing IoT devices: A novel approach using blockchain and quantum cryptography. Internet of Things, 25: 101019. https://doi.org/10.1016/j.iot.2023.101019
[29] Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M. (2018). Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive.
[30] Chen, T., Lu, H., Kunpittaya, T., Luo, A. (2022). A review of ZK-SNARKs. arXiv preprint arXiv:2202.06877. https://doi.org/10.48550/arxiv.2202.06877
[31] Gong, Y., Jin, Y., Li, Y., Liu, Z., Zhu, Z. (2022). Analysis and comparison of the main zero-knowledge proof scheme. In 2022 International Conference on Big Data, Information and Computer Network (BDICN), Sanya, China, pp. 366-372. https://doi.org/10.1109/BDICN55575.2022.00074
[32] Fong, D.K.Z., Selvarajah, V., Nabi, M.S. (2022). Secure server storage based IPFS through Multi-Authentication. In 2022 International Conference on Advancements in Smart, Secure and Intelligent Computing (ASSIC), Bhubaneswar, India, pp. 1-7. https://doi.org/10.1109/ASSIC55218.2022.10088338
[33] Boumezbeur, I., Zarour, K., Keddari, D. (2024). Secure EHR sharing using blockchain and IPFS. Studies in Science of Science 42(7): 1-14.
[34] Khudhur, N., Fujita, S. (2019). Siva-The IPFS search engine. In 2019 Seventh International Symposium on Computing and Networking (CANDAR), Nagasaki, Japan, pp. 150-156. https://doi.org/10.1109/CANDAR.2019.00026
[35] Singh, S., Chakraverty, S. (2022). Implementation of proof-of-work using ganache. In 2022 IEEE Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI), Gwalior, India, pp. 1-4. https://doi.org/10.1109/IATMSI56455.2022.10119271
[36] Qureshi, J.N., Farooq, M.S., Ali, U., Khelifi, A., Atal, Z. (2024). Exploring the integration of blockchain and distributed DevOps for secure, transparent, and traceable software development. IEEE Access, 13: 15489-15502. https://doi.org/10.1109/ACCESS.2024.3509036
[37] Facebook. Facebook/Winterfell: A stark prover and verifier for arbitrary computations. GitHub. https://github.com/facebook/winterfell.
[38] Elibensasson. Elibensasson/libSTARK: A library for zero knowledge (ZK) scalable transparent argument of Knowledge (Stark). GitHub. https://github.com/elibensasson/libSTARK.
[39] El-Hajj, M., Oude Roelink, B. (2024). Evaluating the efficiency of zk-snark, zk-stark, and bulletproof in real-world scenarios: A benchmark study. Information (Switzerland), 15(8): 463.

https://doi.org/10.3390/info15080463
[40] McCabe, C., Mohideen, A.I.C., Singh, R. (2024). A blockchain-based authentication mechanism for enhanced security. Sensors, 24(17): 5830. https://doi.org/10.3390/s24175830
[41] Verma, R., Dhanda, N., Nagar, V. (2022). Application of truffle suite in a blockchain environment. In Proceedings of Third International Conference on Computing, Communications, and Cyber-Security: IC4S 2021. Singapore: Springer Nature Singapore. Ghaziabad, India, Springer: Berlin/Heidelberg, Germany, Springer, Singapore, pp. 693-702. https://doi.org/10.1007/978-981-19-1142-2_54
[42] El-Yahyaoui, A., Omary, F. (2022). An improved framework for biometric database's privacy. International Journal of Communication Networks and Information Security (IJCNIS), 13(3). https://doi.org/10.17762/ijcnis.v13i3.5143
[43] Sangeeta, N., Nam, S.Y. (2023). Blockchain and interplanetary file system (IPFS)-based data storage system for vehicular networks with keyword search capability. Electronics, 12(7): 1545. https://doi.org/10.3390/electronics12071545
[44] Eren, H., Karaduman, Ö., Gençoğlu, M.T. (2025). Security challenges and performance trade-offs in on-chain and off-chain blockchain storage: A comprehensive review. Applied Sciences, 15(6): 3225. https://doi.org/10.3390/app15063225