High-Confidence Computing 4 (2024) 100239

Contents lists available at ScienceDirect

High-Confidence Computing

journal homepage: www.sciencedirect.com/journal/high-confidence-computing

Review article

SoK: Decentralized storage network

Chuanlei Li, Minghui Xu *, Jiahao Zhang, Hechuan Guo, Xiuzhen Cheng [1]

School of Computer Science and Technology, Shandong University, Qingdao 266237, China

ARTICLE INFO

Article history:
Received 16 February 2024
Revised 25 March 2024
Accepted 18 April 2024
Available online 22 May 2024

Keywords:
Decentralized Storage Network
Blockchain
Proof of Storage
Consensus algorithm

ABSTRACT

Decentralized Storage Networks (DSNs) represent a paradigm shift in data storage methodology, distributing and housing data across multiple network nodes rather than relying on a centralized server or data center architecture. The fundamental objective of DSNs is to enhance security, reinforce reliability, and mitigate censorship risks by eliminating a single point of failure. Leveraging blockchain technology for functions such as access control, ownership validation, and transaction facilitation, DSN initiatives aim to provide users with a robust and secure alternative to traditional centralized storage solutions. This paper conducts a comprehensive analysis of the developmental trajectory of DSNs, focusing on key components such as Proof of Storage protocols, consensus algorithms, and incentive mechanisms. Additionally, the study explores recent optimization tactics, encountered challenges, and potential avenues for future research, thereby offering insights into the ongoing evolution and advancement within the DSN domain.

1. Introduction

Cloud storage has become an essential solution for managing the rapidly growing data needs of individuals and organizations. Nowadays, reliable cloud storage providers like Amazon S3 [1], Google Cloud [2] and Alibaba Cloud [3] offer STorage-as-a-Service (STaaS) to users. Cloud storage eliminates the need for localized servers and instead uses remote data centers operated by third-party service providers to store data. This enables users to access their data seamlessly from any device with an internet connection, offering convenience in storing vast amounts of information.

DSNs offer a complementary approach to traditional cloud storage systems, addressing their inherent limitations. Presently, cloud storage relies on a restricted set of providers for both storage and computational resources, without the ability to harness idle resources from everyday users. To expand the capacity of available resources, regular users must contribute their dormant resources to the network. Through this collective contribution, the pooled resources of these ordinary users can significantly augment the network's capabilities. For instance, the combined computational capacity of microcontrollers within a smart home environment can rival that of a single personal computer [4]. The aggregation of such resources through DSNs can effectively supplement conventional cloud storage solutions.

In the era of Web 3.0, there is an increasing demand for users to maintain full autonomy over their data. However, in conventional cloud storage systems, data ownership rests with the providers, thereby relinquishing users' control over their own data. Users are granted only limited permissions to access and utilize their data, resulting in a loss of data sovereignty [5].

DSNs combine storage resources offered by multiple independent storage providers to form a global decentralized file system, without the need for centralized management. In DSNs, actions are publicly traceable, and anyone can participate by either renting out their unused disk space and bandwidth or purchasing decentralized storage services. Blockchain technology acts as a manager and incentive layer in DSNs, encouraging node participation and ensuring file storage consistency even with Byzantine nodes. This is facilitated by DSNs sending cryptocurrency to reliable storage providers and penalizing those with malicious behavior. DSNs also use a Proof of Storage (PoS) scheme to verify if storage providers offer reliable storage services. Storage providers submit storage proofs, which are then verified by clients or smart contracts. Failure to submit the correct storage proof on time indicates a faulty node, and penalties are imposed accordingly. DSNs, such as Sia [6], Storj [7], Swarm [8] and Filecoin [9], are being applied in various fields, including Web 3.0, video streaming, and healthcare, playing their unique roles.

In this paper, we conduct a survey of the existing DSNs projects and papers. In summary, our contributions are as follows:

* Corresponding author.
E-mail address: mhxu@sdu.edu.cn (M. Xu).

[1] Given her role as Editor-in-Chief of this journal, Xiuzhen Cheng had no involvement in the peer-review of this article and had no access to information regarding its peer-review. Full responsibility for the editorial process for this article was delegated to Dr. Zhipeng Cai.

https://doi.org/10.1016/j.hcc.2024.100239

- This paper undertakes an in-depth examination of the developmental path of DSNs, with a specific focus on critical elements including PoS schemes, consensus algorithms, and incentive mechanisms.
- We present an abstract model for DSNs along with a generalized PoS scheme, elucidating the distinctions between various PoS schemes and classifying them into Transitory Storage Proof and Continuous Storage Proof categories.
- Furthermore, we investigate recent strategies for optimization, challenges faced, and potential directions for future research, thus providing insights into the continuous evolution and progress within the DSN domain.

The rest of the paper is structured in the following manner. Section 2 provides a review of the background of DSNs. In Section 3, we present the fundamental model of DSNs. In Sections 4–6, we discuss the current DSNs from three different perspectives: proof of storage, consensus algorithm, and incentive mechanism. Section 7 focuses on the improvements in the performance and security of DSNs. Section 7 presents practical DSN applications. Finally, Sections 9–10 summarizes the existing challenges faced by DSNs and concludes this paper.

2. Background

In this section, we review three technologies that emerged prior to DSN, such as cloud storage, Peer-to-Peer storage, and the InterPlanetary File System.

2.1. Cloud storage

Cloud storage is a technology that enables users to store their data on remote servers located in the cloud. With cloud storage, users can easily expand or shrink their storage capacity (known as elastic cloud) based on their requirements and only pay for the storage space and services they consume. This offers great flexibility to users who can avoid overpaying for unused storage space. Cloud storage providers maintain large data centers worldwide to build a distributed architecture providing high performance and fault tolerance. Cloud storage typically features centralized managers responsible for monitoring and coordinating the entire system. When you use a cloud service to store or host your data, you are essentially giving control of your files to the provider, even though you still own them [10].

There are three types of cloud storage based on different usage scenarios [11]: object storage, file storage, and block storage. Object storage is designed for storing unstructured data like photos and videos. File storage stores data in a hierarchical folder and file format, and is often used in directories and data repositories. Block storage breaks data into blocks and stores them in the most efficient location for the system, allowing for quick retrieval. Block storage is ideal for handling large volumes of data with low latency demands, such as databases and workloads that require high performance.

2.2. Peer-to-peer storage

Peer-to-Peer (P2P) storage is a technique that uses the storage capacity of individual nodes within a P2P network to create a shared storage pool for storing and sharing data. In a P2P storage network, each node can directly interact with other peers, eliminating the necessity for centralized management. Every node has similar privileges and responsibilities. Each connected device can act both as a client and a server, offering storage capacity and computing resources while consuming resources provided by other peers. Retrieving data from multiple peers simultaneously is made convenient and efficient with P2P storage networks. These networks offer data redundancy, which ensures uninterrupted access to data even when peers become unavailable. This means that users can still access their data even if some of the peers go offline.

FreeNet [12] and Ivy [13] are P2P storage projects. Communication between FreeNet nodes is encrypted and routed through other nodes to make it difficult to trace. Ivy is a writable P2P storage system, each participant maintains a set of logs that record modifications made to the system, and periodically synchronizes these logs through snapshot scans. To optimize the P2P storage system, Giroire et al. [14] proposed a data placement strategy to reduce bandwidth consumption and [15,16] give suitable management methods to keep the system robustness.

2.3. InterPlanetary file system

The InterPlanetary File System (IPFS) is a content-addressable peer-to-peer network that provides distributed data storage and delivery [17]. The network is built around the innovation of content addressing. It stores, retrieves, and locates data based on the content identifier (CID) of actual content of data rather than the name or location.

When a file is added to the IPFS network, it first gets divided into multiple smaller chunks, each being 256KB by default. Each chunk is then assigned a unique CID, which is used as a leaf node to create a Merkle Directed Acyclic Graph (DAG). The hash function used to create CIDs ensures that no cycles are created during the process. The root of the Merkle DAG is used as the CID of the complete file, used to retrieve the file. Merkle DAG differs from a Merkle tree in that it does not enforce balance, and a node can have multiple parents. In IPFS, Merkle DAG is also used for file version control. Since the updated file and the original file share the same Merkle subgraph, only the difference between versions needs to be stored instead of the entire Merkle DAG. This is due to the duplicate data chunks forming the same Merkle subgraph.

In IPFS, each peer is identified by a unique 256-bit PeerID, which is as long as a CID. To publish content, a peer that stores a data chunk generates a provider record and a peer record, and then pushes them into the distributed hash table (DHT). The provider record maps the CID to the PeerID, while the peer record maps the PeerID to the address that is used to find the actual location. The updated DHT is synchronized with the twenty closest peers in terms of their PeerIDs' XOR distance from the CID [18]. When a peer retrieves a file with its CID, it performs multi-round iterative lookups. The retrieval request is first forwarded to three peers whose PeerIDs are closest to the CID in the requesting peer's routing table. The peers that have the requested content directly return the content and then end the lookup. If they have a copy of the provider record related to the CID, they return the PeerID and the peer address if they have it. Otherwise, the next round of lookup is performed. Once the target peer is found, the requesting peer establishes a connection with it and transmits data through the Bitswap protocol [19].

Daniel et al. [20] conducted detailed research on P2P storage related to IPFS and summarized that IPFS combines ideas from BitTorrent, Kademlia, Git, and information-centric networking (ICN) to create a new data network. To improve IPFS, IPFSz [21] was proposed to solve the storage space-saving issues. Sun et al. [22] leverage blockchain to provide access control in IPFS which can improve system security.

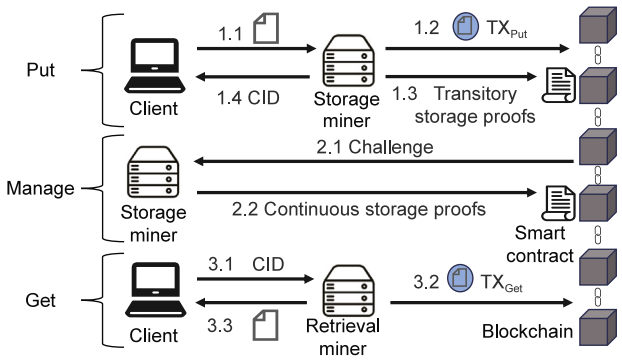

Fig. 1. DSN Model.

3. DSN model

In this section, we present an abstract model of DSNs, encompassing its distinct components and functional design. Through a detailed examination of these key aspects, we aim to unravel the inner workings of DSNs. A DSN is operated by clients and miners. Clients provide a friendly means for users to interact with miners. It is responsible for data uploading and retrieving. For DSNs that store ciphertext, clients also have the task of encrypting and decrypting data. Miners are divided into two types based on their different tasks, namely storage miners and retrieval miners. Storage miners provide storage resources to clients and have the ability to mine new blocks to receive mining rewards. If users want to be storage miners, they must pledge their storage space to the network to ensure reliable service. Retrieval miners participate in the network by offering data retrieval services to clients. They do not need to pledge storage and generate storage proofs. It is natural for storage miners to be competent as retrieval miners.

In DSNs, the various operations carried out by clients and miners can be summarized into three primary functions, namely Put, Manage, and Get.

DSN = (Put, Manage, Get).

- Put(D, SM) $\rightarrow$ CID: Clients execute this function to upload the data D to a storage miner SM to store data in DSNs, and obtain the content identifier CID.
- Manage(D, SM, ReM): This function runs by the storage miners and retrieval miners to manage data D. The function ensures the available storage, audits the storage services, and resists potential faults.
- Get(CID, ReM) $\rightarrow$ D: Clients execute this function to upload the content identifier CID to a retrieval miner ReM, and get corresponding data from it.

Fig. 1 presents the DSN model to illustrate how the primary functions work. A client first uploads a file which is then divided into smaller data chunks. These chunks are then sent to multiple storage miners. At this point, storage miners create a put transaction, TX_{Put}, which is then broadcast to the blockchain network. The storage miner then generates the transitory storage proofs to demonstrate that it has correctly stored the data. During the process of data management, nodes can call the Manage function to verify the continuous storage proofs, allowing for control over available storage and the ability to repair any potential faults.

In the event of data retrieval, a client forwards a retrieval request containing the CID to retrieval miners. These retrieval miners retrieve the corresponding data fragments and reconstruct the original file. Analogous to the storage procedure, a retrieval miner initiates a transaction, denoted as TX_{Get}, which is then broadcasted across the blockchain network. It is imperative to emphasize that the completion of both the Put and Get functions is contingent upon the confirmation of transactions on the blockchain.

There are three key techniques for DSNs: proof of storage schemes, consensus algorithms, and incentive mechanisms.

- **Proof of Storage.** PoS is one of the innovations of DSNs that distinguishes it from other storage approaches. In a decentralized network, nodes cannot trust each other. Therefore, a trusted mechanism must be established to verify that a particular node is indeed providing storage space and not lying. This is crucial for ensuring the integrity and security of the decentralized network. Without such assurance, we cannot deploy resource scheduling algorithms, as the availability of resources is unknown. This problem applies to more than just storage proofs. In the future, a general protocol is needed to prove computational resources and bandwidth as well.
  There are two types of storage proofs: transitory storage proofs and continuous storage proofs. Once a client uploads data, storage miners are required to generate transitory storage proofs to confirm that they have indeed stored the data. During storage maintenance, the storage miners must provide continuous storage proofs to demonstrate that they have correctly stored the data for a specific period. To achieve this, the storage miner reads a random number from the blockchain as a challenge, uses this number as an input to generate the corresponding storage proof, and sends it to the smart contract for verification. Smart contracts and blockchain technology serve as a trusted third-party.
- **Consensus Algorithm.** DSNs require that all participants have a fair chance of contributing and benefiting from the network based on their contributions. All participants need to reach a consensus on the state of the system, which is actually achieved by selecting an honest miner (with high probability) recognized by the participants to package transactions and mine the next block in the blockchain. A robust consensus algorithm helps DSN ensure the integrity of its network and prevent data tampering.
- **Incentive Mechanism.** DSNs are open and self-coordinating systems that ensure normal operation through incentive mechanisms. An incentive mechanism involves the use of cryptocurrency rewards for miners who allocate their storage space and bandwidth to store and retrieve data. However, miners who engage in malicious behaviors face punishment.

Comparison of DSN and Cloud Storage. To elucidate the distinction between DSN and conventional cloud storage systems, we conduct a comparative analysis across seven dimensions, as illustrated in Table 1. DSN, leveraging blockchain technology, integrates Byzantine fault-tolerant consensus algorithms and cryptocurrency mechanisms. In contrast to cloud storage, DSN requires PoS protocols to guarantee data availability, along with incentive mechanisms to incentivize participation and resource aggregation among individual users. *DSN vs Blockchain.* While both blockchain and decentralized storage networks leverage decentralization and cryptographic principles, they serve distinct purposes: blockchain primarily focuses on transaction recording and consensus, while decentralized storage networks focus on securely storing and accessing data in a distributed manner.

4. Proof of storage

4.1. Overview

PoS scheme is the most significant part of DSNs. It is used to check whether the storage miners have correctly stored data. In

Table 1
The comparison of decentralized and centralized storage systems.

Storage strategy	Current work	Proof of storage	Ledger structure	Consensus algorithm	Smart contract	Incentive	Redundancy	Cryptocurrency
Decentralized	Sia [6]	Merkle tree	Chain	PoW	Yes	Yes	Erasure code	Siacoin
	Storj [7]	PoR	Chain	PoS❶	No❷	Yes	Erasure code	STORJ
	Filecoin [9]	PoRep/PoSt	DAG	EC❸	Yes	Yes	Copy data	FIL
	FileDAG [23]	PoRep/PoSt	DAG	DAG-Rider❹	Yes	Yes	Copy data	FIL
	Swarm [8]	Merkle tree	Chain	PoW	Yes	Yes	Erasure code	BZZ❺
Centralized	Amazon S3 [1]	✘	✘	Paxos	No	No	Copy data❻	✘
	Google Cloud [2]	✘	✘	Paxos	No	No	Erasure code	✘
	Alibaba Cloud [3]	✘	✘	Paxos	No	No	Copy data	✘

❶Proof of Stake in Ethereum.
❷Smart contract storage remains a long term goal for Storj, according to Storj forum.
❸Expected consensus.
❹DAG-Rider is an asynchronous Byzantine Atomic Broadcast protocol.[24]
❺BZZ in Swarm is used to settle excess debt due to unequal bandwidth consumption and purchase the storage services.
❻The redundant file objects are stored on multiple devices in discrete data centers.
✘ means this method does not have this feature.

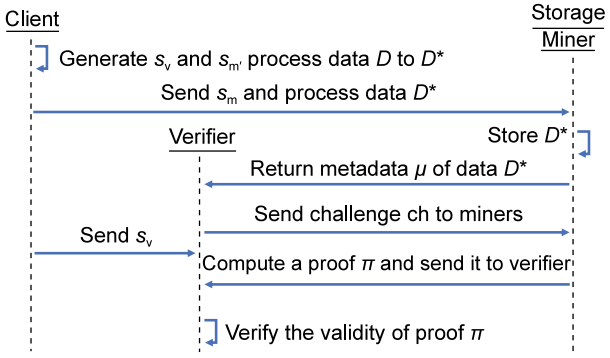

Fig. 2. The procedure of executing PoS.

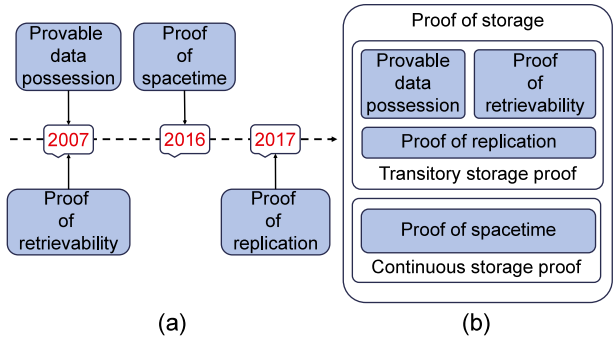

Fig. 3. Timeline and relationship of PoS schemes. (a) Timeline. (b) Relationship..

order to ensure data security in the public environment, the basic principle of data verification should comply with the following description: clients need to be able to verify that a storage miner has retained file data without retrieving the data from the storage miner. A generalized PoS scheme can be described as follows.

$\mathsf{PoS} = (\mathsf{Setup}, \mathsf{Store}, \mathsf{Prove}, \mathsf{Verify})$

- $\mathsf{Setup}(1^\lambda, D) \rightarrow (S_v, S_m, D^*)$: Clients leverage this function to generate S_v and S_m based on a security parameter λ, where S_v and S_m are scheme-specific variables for verifier and storage miner respectively. Client processes data D to D^*, and the processing method varies depending on the specific PoS scheme.
- $\mathsf{Store}(D^*) \rightarrow \mu$: Storage miners store the processed data D^* sent by client, then calculates the corresponding metadata μ and returns it to the verifier.
- $\mathsf{Prove}(S_m, \mathsf{ch}, D^*) \rightarrow \pi$: Storage miners get a challenge ch and generate the storage proof π to show they correctly store processed data D^*.
- $\mathsf{Verify}(S_v, \mu, \mathsf{ch}, \pi) \rightarrow \{0, 1\}$: Verifier utilizes this function to check the validity of storage proof π. Verifiers can be assumed by different parties such as clients or smart contracts.

To elucidate the operational mechanics of a PoS system, we delineate its procedural flow in Fig. 2 through a straightforward scenario where a client, who also assumes the role of a verifier, initiates a challenge to a storage miner. Denoting S_m and S_v as a key pair pk, sk, the client locally processes data D into D^*, which is subsequently transmitted to storage miners. These processing techniques may encompass encryption and erasure coding. The client transmits S_m along with the processed data D^* to the storage miner, who store D^* and compute its metadata μ before returning it to the client. To conduct audits on storage integrity, the client, functioning as a verifier,[2] dispatches a randomized challenge ch to the storage miner. The storage miner then computes a storage proof π based on the challenge and the actual stored files, subsequently forwarding π to the client. Ultimately, the client verifies the storage proof.

Fig. 3 shows the current prevailing methods for generating storage proofs. We display their timeline and the relationship between them. Based on whether the storage proofs are time-related, we classify PoS schemes into two categories: transitory storage proof and continuous storage proof. Typically, a DSN employs a combination of these two types of PoS.

4.2. Transitory storage proof

Transitory storage proof refers to the storage proof that does not have time as an input during proof generation. For example, in DSN.Put function, after storage miners receive and store the data, they send storage proofs to smart contract to show that they have correctly stored the data.

Provable Data Possession/Proof of Retrievability. The Provable Data Possession (PDP) [25] protocol allows users to check the integrity and availability of their outsourced data on untrusted data servers. A client processes the file that needs to be stored and generates the metadata of the file. For verification, client uses a random challenge to require the server to respond the

[2] In the context of DSNs, a verifier may denote the client, blockchain, or other third-party auditors.

proof of possession of specific data chunk. Client can leverage the metadata to verify the proof.

In Proof of Retrievability (PoR) [26], after a client processes the file to blocks, it adds redundancy check blocks which they have called sentinels. Verifier challenges storage miner by requesting them to return the sentinel value at specified locations. If storage miners delete a portion of file, it is highly possible that they lack sentinels and cannot respond correctly. One of the common methods for generating sentinels is erasure code [27]. This sentinel design ensures the system has the ability to retrieve and fix files that have small file corruption. The core idea of PDP and PoR is to verify the random data chunks. PoR is similar to PDP. For brevity, we can call them the PDP/PoR scheme [28].

Sia [6] is a representative of early DSNs projects. It enables the creation of file contracts between client and storage miners. The file contracts contain various relevant information for file storage, and storage proof is one of these information. Following the idea of verifying random data chunks in PDP/PoR scheme, storage miners in Sia network demonstrate their correct storage by offering a chunk of the original file alongside a list of hashes sourced from the file's Merkle tree. If they prove the possession of a random chunk, it is highly probable that they are storing the entire file. The contracts are written in a blockchain, which makes them publicly auditable.

Storj [7] of the same period as Sia, is also based on the PDP/PoR scheme and leverage Merkle tree to obtain the storage proofs. If a storage miner provides invalid storage proofs, it is marked as a bad miner and the corresponding data is redistributed to a new storage miner. Unlike Sia, in addition to client and storage miner, Storj also introduces another type of node called satellite. Satellites act as the mediator between clients and storage miners, facilitating the storage interaction. They are responsible for metadata storage, node discovery, data auditing and data repair [29]. However, their security decreases due to the untrustworthiness of the third party.

Proof of Replication. Filecoin introduced Proof of Replication (PoRep) [30]. Different from PDP/PoR, PoRep enables the storage miners to convincingly demonstrate to a client that specific data has been successfully replicated onto distinct and dedicated physical storage space. It guarantees the availability of redundant copies and resists tampering and Sybil attacks. If a DSN does not support PoRep, a malicious node can pretend to store multiple replicas while actually storing only one to earn extra benefits. When generating a proof, PoRep puts each data replica as one of the inputs of function PoS.Prove, so that verifier can check if a storage miner has stored the replica via the proof.

Filecoin [9] is a popular DSN built on top of IPFS. In Filecoin, a Seal-based PoRep proves storage miners are storing independent replicas. The Seal protocol mandates that storage miners produce a duplicate of the original data utilizing a collision-resistant encoding function, wherein the miner's identity serves as a parameter for the encoding function to ensure distinctness among replicas, thereby establishing physical independence. A parameter denoted as k, indicative of the number of encoding iterations, is employed to impede the Seal protocol. The iterative execution time of the encoding function k times typically exceeds the duration of the conventional challenge-prove-verify process by a factor ranging from 10 to 100 [30]. The slow property prevents storage miners from deleting the data replica and regenerating it when they are required to generate storage proofs. Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (zk-SNARKs) [31] plays a vital role in the real implementation of PoRep in Filecoin. The storage proof based on zk-SNARKs is non-interactive and publicly verifiable. zk-SNARKs improves the on-chain verification efficiency by reducing storage proof size and verification time.

Damgard et al. [32] present a rigorous formal treatment of PoRep. Cecchetti et al. [33] introduce Public Incompressible Encoding (PIE), a practical and efficient tool for PoRep. PIE operates within a public DSN setting, where data uploads are unencrypted, and redundancy verification is public. This approach effectively thwarts undetectable malicious data compression. Yuan et al. [34] propose a method that integrates incompressible encoding with a homomorphic linear authenticator to audit data replication integrity without relying on timing assumptions.

4.3. Continuous storage proof

Utilizing the transitory storage proof scheme, clients can ensure the presence of data as of the instant the storage miner generates the proof. The issue lies in the fact that they are unable to ensure consistent availability of data, as stated in [35]. For instance, a storage miner might agree to store data only when receiving a file and generating a transitory storage proof. Afterwards, they could try to avoid storing the data by deleting it or outsourcing it to other storage miners. To prevent such malicious behavior, we need to use time as a parameter for calculating continuous storage proofs.

Frequent Storage Proof. On the basis of existing transitory storage proof schemes, one of the methods to achieve continuous storage proof is to require storage miners to send storage proof every certain time interval. In other words, the storage miners are required to submit storage proof frequently throughout the data storage duration.

The file contract in Sia records the challenge frequency and the maximum tolerable number of lost storage proofs. The challenge frequency specifies the time interval for storage miners to generate storage proofs. If the correct proofs can be submitted within the specified time interval, the reward will be automatically sent to the payment addresses recorded in the file contract. If the number of unsuccessful submissions of storage proofs exceeds the predefined threshold, the file contract will be invalidated.

Proof of Spacetime. Moran et al. [36] first proposed Proof of Spacetime (PoSt) in 2016. PoSt requires the storage miners to show that they possess and maintain data chunks in a period of time, demonstrating their continuous participation in the network. It can discourage storage miners from only submitting transitory storage proofs. PoSt additionally chooses a time parameter t as one of the inputs of function PoS.Prove, so that the verifier can check if storage miner store data in a period of time. Filecoin introduces a PoSt scheme which is similar to an iterative PoS. Storage miners carry out the PoS protocol several times, with the outcome of the previous round being used as a seed to generate a new challenge in the next round. This means that if the proof in the previous round is incorrect, the results of subsequent rounds will also be incorrect. This structure can be viewed as a proof chain. Early research on PoSt such as [36] only guarantees the usage of space resources, but cannot ensure data retrievability. Ateniese et al. [37] formally organized the new PoSt scheme which has efficient data availability. StoRNA [38] addresses the problem of statefulness and non-transparency in PoSt generation. Statefulness means verifier can only run a limited number of interactions. StoRNA enables non-interactive continuous availability of outsourced data in a publicly verifiable way. Besides, ePoSt [35] minimizes the cost of the client while ensuring statelessness (opposite to statefulness) and public verifiability. Combining the features of PoRep and PoSt, Proof of Replicated-time (PoRt) [39] makes the client believe that storage miners are storing the replica of same data for a period of time, ensuring continuous availability of replicated storage.

5. Consensus algorithm

Consensus mechanisms play a fundamental role in establishing a globally consistent viewpoint across network nodes. Within the realm of blockchain technology, Byzantine Fault-Tolerant (BFT) consensus protocols hold particular significance, ensuring unanimity regarding transaction validity and sequencing despite the presence of Byzantine adversaries. Given that DSNs operate on blockchain infrastructures, they are heavily reliant on these consensus mechanisms as foundational elements. While a detailed exploration of blockchain technology is beyond the scope of this discourse, interested readers are encouraged to consult recent comprehensive surveys on the subject [40]. However, it is imperative to differentiate between consensus mechanisms and PoS, the latter being devised to authenticate the presence and integrity of stored data. PoS endeavors to furnish evidence attesting to the correct storage and retrievability of specific data chunks. Blockchain technology records various transactions, facilitating rigorous proof verification within DSNs, thereby rendering proofs amenable to public scrutiny.

Consensus algorithms in blockchain can be divided into two categories: Proof-of-X (PoX) consensus and BFT consensus. Where PoX refers to a set of protocols used in blockchain to determine which consensus node should act as the leader to propose a block. BFT consensus is to ensure that a distributed system can withstand system failures by reducing the impact of byzantine nodes and preserving the consensus reached by the honest majority.

Proof of Work. Proof of Work (PoW) is a form of cryptographic proof that shows a miner has already expended a certain amount of computational effort. Sia uses Siacoin as its cryptocurrency. Siacoin is implemented as an altcoin, which means it is a derivative of Bitcoin. Sia broadens transaction parameters by adding elements such as file contracts and storage proofs. The consensus algorithm in Siachain is PoW, which is as same as Bitcoin. In PoW, miners (or hosts in Sia's context) compete to solve complex mathematical puzzles to show they indeed invest a certain amount of computational power. The first miner to solve the puzzle gets the right to add the next block of transactions to the blockchain and is rewarded with Siacoins.

Proof of Stake. Proof of Stake is firstly leveraged in Peercoin [41]. It selects miners in proportion to their quantity of cryptocurrency holdings rather than the contributed computational effort. Storj [7] relies on the Satellites. The payments between Satellites and miners are via the Ethereum-based ERC20 [42] STORJ token. So the consensus algorithm in Storj is Proof of Stake. Miners who want to participate in consensus process provide a certain amount of tokens as a pledge to show their working ability and get corresponding stake. Then system randomly selects the miner who can create the next block based on their pledged quantity. Usually, the more the pledge, the higher the chance of being chosen to create the next block. Compared to the huge computational power consumption of PoW, this protocol succeeds in reducing energy expenditure to negligible levels [43].

Expected Consensus. Expected Consensus (EC) is a probabilistic Byzantine Fault-Tolerant consensus protocol developed by Filecoin. This protocol mainly consists of three procedures: leader election, mining blocks and fork selection. The overall process is as follows: participants check whether they have been elected as leaders, and if they are qualified, they will get the right of mining blocks and choose a fork that meets the requirements to continue mining blocks downwards. Fig. 4 shows the process of EC.

Blocks in Filecoin are ordered by epoch. Epoch is a time period of thirty seconds in length [44]. At the beginning of every epoch, participants run the leader election.

$\mathsf{Election} = (\mathsf{ProveElect}, \mathsf{VerifyElect})$

- $\mathsf{ProveElect}(r, t, \mathsf{SM}) \rightarrow \{0, \pi\}$: This function executes by storage miner SM and calculates whether the following inequality holds true: $\mathsf{H}(\langle t \parallel r \rangle_{\mathsf{SM}})/2^L \leq \frac{q}{Q}$, where r is a public randomness available that can be extracted from the blockchain at epoch t, $\mathsf{H}()$ is a secure hash function with a result length of L, $t \parallel r$ can be signed by SM and return $\langle t \parallel r \rangle_{\mathsf{SM}}$. Q is the total amount of storage resources across the entire network, and q is the storage resource provided by SM at epoch t. They can be queried on public tables. If the inequality holds true, return the leader proof $\pi = \langle t \parallel r \rangle_{\mathsf{SM}}$, otherwise return 0.
- $\mathsf{VerifyElect}(\pi, t, \mathsf{SM}) \rightarrow \{0, 1\}$: Network node executes this function to check if π is a valid signature from SM and check if q is the storage resources of SM at epoch t. Then it calculates if $\mathsf{H}(\pi)/2^L \leq \frac{q}{Q}$ and returns a bit $x \in \{0, 1\}$ to show the validity of the leader.

The leader election is unpredictable due to the presence of random number and no one knows who is the leader until they broadcast the leader proof for all participants to verify. The probability of winning an election is proportional to each miner's assigned storage, which makes it fair.

The elected miner utilizes an unbiased randomness generator to obtain a random value, then uses this value to determine the sector in which they must generate a WinningPoSt. Correct WinningPoSt demonstrates that the elected miner keeps a sealed copy of the data before the end of the current epoch [45]. If the miner is unable to generate the correct WinningPoSt within the specified time, it will be considered as a unqualified miner who cannot proceed to the next step of mining a block. Due to the security of the PoSt in Filecoin, byzantine miners cannot become leaders. A storage miner who has submitted the correct election proof and WinningPoSt is qualified to mine a block and include the election proof and the WinningPoSt for public verification. The leader broadcasts this successfully mined block to the network, and participants verify the validity of this block and add it to their own block ledger if it is valid.

The system possesses the capability to adjust specific parameters in order to fine-tune the anticipated quantity of elected leaders within each epoch, thereby delineating the notion of expected consensus. It is possible for certain epochs to experience either a lack of leaders or an abundance thereof. In the absence of a leader, an empty block is appended to the chain during said epoch. At the culmination of every epoch, the chain undergoes expansion through the addition of one or multiple blocks. Consequently, the structure of the block ceases to adhere to a linear chain; rather, it manifests as a DAG, known as the chain of tipsets within the context of Filecoin. A tipset denotes a non-empty collection of blocks sharing common parentage and mined within the same epoch. For instance, within Fig. 4, the set $\{E, F\}$ represents a tipset occurring at epoch $n + 1$, distinguished by its shared parents $\{B, C\}$. In order to reflect the cumulative computational effort expended, or in the case of Filecoin, the aggregated volume of committed storage, the fork selection mechanism is devised to favor the heaviest chain, diverging from the simplistic preference for the longest chain akin to Bitcoin's protocol. The weight attributed to a chain in Filecoin correlates with both the quantity of blocks and the cumulative storage capacity committed to said chain. In Fig. 4, we exclusively consider the block count as indicative of weight; accordingly, the chain comprised of red tipsets attains the greatest weight, thereby emerging triumphant and being recognized as the principal chain. Ultimately, the novel valid block becomes interlinked with the principal chain.

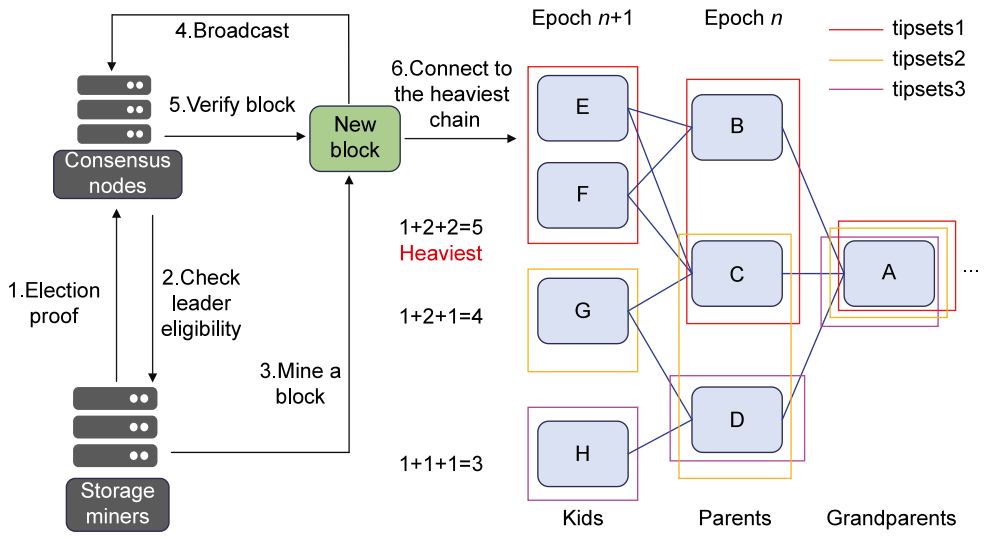

Fig. 4. Expected consensus.

6. Incentive mechanism

DSNs operate in a public environment, the incentives are employed to maintain the normal interaction of network. Generally speaking, incentives serve to reward the miners who follow the rules while penalizing the miners who engage in malicious behavior.

Monetary Incentives. DSN can directly leverage cryptocurrency to construct monetary incentives due to its incorporation of blockchain technology. Participants receive cryptocurrency rewards by providing storage or retrieval services and mining blocks in blockchain. Conversely, malicious participants cannot receive rewards or even face monetary penalties. Each distinct DSN boasts its individual native cryptocurrency: Siacoin within Sia, STORJ within Storj and FIL in Filecoin. Siacoin and FIL are paid to participants through smart contracts. This automatic and transparent payment method can prevent human fraud and reduce the involvement of intermediaries to lower costs. The cryptocurrencies constitute the foundation upon which their respective monetary incentive mechanisms.

Reputation Incentives. Another incentive mechanism, reputation incentive, operates independently of cryptocurrencies, instead leveraging reputation value. Participant benefits are directly influenced by the reputation value they hold. This mechanism plays a role in identifying and incentivizing trustworthy and advantageous participants across the network.

Storj's reputation system encompasses four distinct subsystems: the proof of work identity system, the initial vetting process, the filtering system, and the preference system. Within the proof of work identity system, newly onboarded storage nodes are mandated to undergo a proof of work procedure as part of their identity generation, aimed at mitigating potential Sybil attacks. During the initial vetting process, the satellite initially permits unvetted storage nodes to store non-critical data, leveraging erasure codes to safeguard data integrity in the event of loss. Upon accruing a sufficient volume of data over a designated timeframe (typically spanning at least one payment period), these storage nodes attain qualified and trusted status. In the filtering system, nodes demonstrating aberrant behavior, such as inability to furnish storage proof or data retrieval, incur a decrement in their reputation score. Should a node's reputation fall below a predefined threshold, it necessitates re-entry into the initial vetting process. Subsequently, within the preference system, following the exclusion of disqualified storage nodes, the remaining qualified nodes are prioritized based on statistical metrics gleaned during the audit process, including throughput and latency. Storage nodes allocated higher priority are more apt to be tasked with storing newly generated data.

Bandwidth Incentives. Swarm [8] has its unique bandwidth incentive mechanism. The core of this mechanism is the Swarm Accounting Protocol (SWAP). Nodes engage in a service-for-service exchange when they forward information. They view service as a currency that can be traded. Concurrently, they record their individual consumption of bandwidth in relation to each of their associated peers. However, once the discrepancy in services provided by both parties surpasses the pre-agreed threshold, the party who has accumulated liabilities faces a choice: they can either opt to await gradual amortization of their liabilities over time, or they can promptly settle the debt by means of payment in BZZ tokens on the blockchain. So basic services are free because a small amount of bandwidth usage can be amortized, also you can quickly reduce the discrepancy between the two parties through payment to enjoy high-quality services.

Lakhani et al. [46,47] evaluate the fairness of incentive mechanisms in Swarm network and conclude that the current parameter settings may not achieve optimal fairness in the distribution of rewards within the Swarm ecosystem. They provide guidance on beneficial parameter settings and propose novel instantiations of these mechanisms and highlight how these may be beneficial for the Swarm ecosystem.

7. Optimization and enhancement approaches

This section focuses on the optimizations in terms of performance and security of DSNs. Firstly, we analyze the redundancy mechanism in DSNs. Then we start with various attacks and introduce the methods to ensure data security in DSNs. Finally, we describe ways to increase data privacy.

7.1. Redundancy reduction

The redundancy ensures that even if nodes fail, the data remain available from other replicas. The simplest redundancy mechanism is to store multiple backup files, which wastes storage space. Erasure code [48] is a common method for reducing data redundancy in DSNs. Sia, Storj and Swarm all leverage erasure code, which breaks original data into smaller chunks and generates additional pieces of information, known as parity chunks.

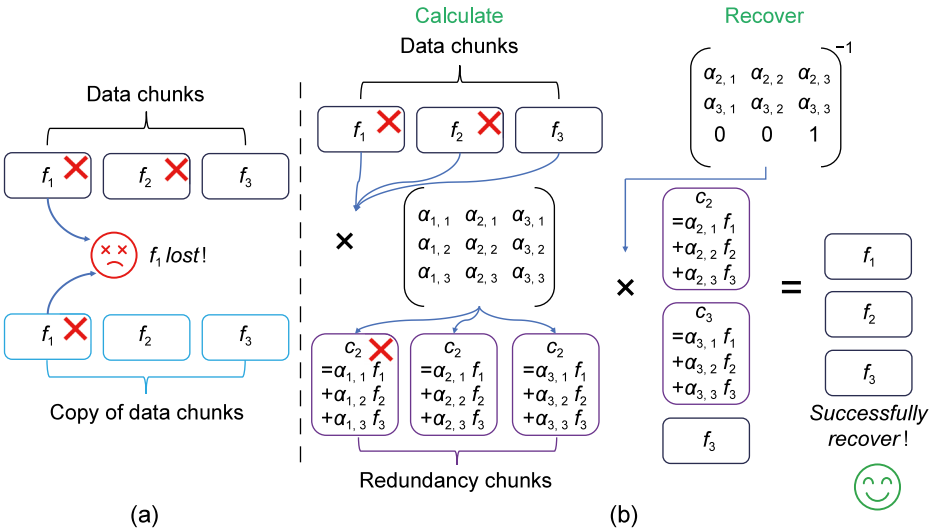

Fig. 5. An example of erasure code. (a) Full replication. (b) Erasure Code.

An erasure code is often described by two parameters, k and n. If data is encoded with a (k, n) erasure code, there are n total chunks, and the original data can be recovered from any subset of k unique chunks. Erasure code usually uses Vandermonde matrix or Cauchy matrix as encoding matrix, which satisfy the following properties: for any $k \times n$ matrix, any $k \times k$ submatrix of it is invertible.

Fig. 5 shows an example of applying erasure code. (a) is the situation of only storing full replicas. If f_1 and its corresponding copy are both lost, f_1 cannot be recovered. (b) describes the scenario of using $(3, 6)$ erasure code. Any 3 of 6 chunks can be used to recover the original data. In this case, the original chunks $\{f_1, f_2, f_3\}$ are transformed into parity chunks $\{c_1, c_2, c_3\}$ through matrix operations. If $\{f_1, f_2, c_1\}$ are lost, we can easily recover the original data $\{f_1, f_2, f_3\}$ through matrix operations with $\{f_3, c_2, c_3\}$. Evidently, using erasure codes instead of full replication provides advantages such as reduced data duplication and ability to recover from failures.

In Storj whitepaper [7], authors give an analysis of storage overhead and security for full replication and erasure code. They construct a Poisson distribution model to calculate the probability that at most $n - k$ chunks of the file are lost in a month and the file can still be rebuilt. The results indicate that erasure codes can achieve higher security with relatively lower storage overhead. Also, it makes the storage miners get more payment, because the erasure code saves storage space and the same payment can be distributed to fewer storage miners. In order to improve the fault tolerance of the network, Storj adds two additional parameters m and o, and $k \leq m \leq o \leq n$. The value of m is the minimum safety value, when the satellite detects that the amount of available chunks is less than m, it will trigger a repair to ensure that there are always k or more chunks available. The o value is set to prevent the long-tail responses in distributed systems. After uploading o chunks to gain enough redundancy, the remaining $n - o$ chunks can be stopped from uploading. This enables the upload to proceed at the pace of the fastest nodes within a set, eliminating the need to wait for the slower nodes. BFT-DSN [49] is a Byzantine fault-tolerant decentralized storage network that employs storage-weighted BFT consensus, erasure coding, and advanced verification techniques to reduce redundancy, ensuring robustness and superior Byzantine resilience compared to existing DSNs.

7.2. Attack and defense

P2P networks and proof systems are crucial components of DSN. Here, we discuss attacks and corresponding defense methods targeting these two modules.

P2P Network. The Sybil attack, as delineated by Douceur [50], poses a pervasive threat within P2P networks, capitalizing on the inherent decentralized architecture of the network. In this strategy, a malicious entity fabricates numerous deceptive nodes, all under its control, establishing a many-to-one mapping of identity to entity. This affords the malevolent actor the capacity to claim the storage of multiple replicas of specific data, ostensibly for corresponding rewards, while in reality, only a singular copy is stored. Filecoin's PoRep mechanism associates the generation of data replicas with node identities, ensuring that the temporal investment required for replica generation from the original data is sufficiently prolonged. Consequently, Sybil attackers are constrained by computational limitations, impeding their ability to generate data replicas and storage proofs within the stipulated timeframe. To avoid reprisals, these nodes are compelled to accurately store data replicas. Storj's storage node reputation system mandates nodes to undergo an extended initial vetting period, preventing fledgling nodes from accessing data from established reputable storage nodes, thereby thwarting Sybil attacks.

Notwithstanding the effectiveness of Storj's reputation system against Sybil attacks, a potential vulnerability, termed the Honest Geppetto attack, is acknowledged in the Storj whitepaper. In this scenario, the attacker orchestrates the operation of multiple puppet nodes within Storj, progressively establishing their trust within the network. Upon achieving a certain level of trust, these nodes defect to malicious behavior. Mitigating this risk necessitates meticulous analysis of storage node correlations and the strategic expansion of the network.

Prunster et al. [51] identified the eclipse attack in the IPFS, primarily constituting a network-centric security threat. In this attack, the assailant strategically isolates a target node or group of nodes, preventing their connection with honest and legitimate nodes. This grants the attacker the ability to manipulate data stored on the isolated nodes, coercing clients into paying ransoms to prevent data leakage or loss. Storj addresses eclipse attacks by employing public key hashes as node IDs, utilizing signatures based on these public keys, and executing multiple disjoint network lookups, adhering to the S/Kademlia protocol [7].

Storage Proof System. Benet et al. [30] systematically investigate the vulnerabilities associated with outsourcing attacks and

generation attacks within DSNs. In the context of outsourcing attacks, adversaries exploit the situation by subcontracting data initially designated for self-storage to alternative storage miners, thus assuming the role of intermediaries and securing advantages without directly bearing the responsibilities of storage. In the case of generation attacks, malicious entities deploy a concise program capable of regenerating the committed data. Upon data retrieval or challenge issuance by the client, this program reconstructs the data along with its corresponding storage proof. Notably, the program's size is often smaller than the data it endeavors to replicate, resulting in storage miners receiving benefits disproportionate to the actual physical storage employed.

To mitigate these vulnerabilities, Filecoin integrates Seal, a mechanism engineered to conspicuously impede attackers by rendering their responsiveness notably slower than that of an honest prover when addressing challenges. This strategic design ensures that attackers are unable to expeditiously acquire the data replica between the receipt of a challenge and the generation of the storage proof, whether through retrieval from outsourced storage miners or regeneration facilitated by the compact program.

7.3. Data privacy protection

Preserving data privacy is of significance within the context of DSNs. In instances where ciphertext storage is employed in DSNs, data undergoes encryption before storage, ensuring that exclusive access and decryption privileges are reserved for authorized entities. The security of stored data hinges upon the utilization of encryption algorithms and cryptographic keys, thereby ensuring that, even in scenarios involving the compromise of miners, the data remains unintelligible without the requisite decryption keys.

In the realm of cloud storage, access permissions are conventionally managed by a centralized platform. Conversely, within DSNs, clients possess the capability to distribute secret keys among users, thereby alleviating concerns associated with untrustworthy centralized platforms. Wang et al. [52] articulate a comprehensive framework that integrates the IPFS, the Ethereum blockchain, and attribute-based encryption (ABE) to establish resilient access control mechanisms. Steichen et al. [53] employ smart contracts to govern the access control list within the IPFS framework.

Proxy re-encryption (PRE) emerges as an advantageous method for cryptographic access control within DSNs. In contrast to encrypting the transmission key, PRE involves the re-encryption of data. This methodology is particularly well-suited for scenarios in which Alice endeavors to share encrypted data with Bob without divulging her secret key. A semi-trusted proxy, armed with a conversion key or re-encryption key generated by Alice based on Bob's public key and her private key, facilitates the transformation of ciphertext encrypted by Alice's private key into ciphertext encrypted by Bob's public key. Following re-encryption, the converted ciphertext becomes amenable to decryption by Bob's private key, thereby effectuating the assignment of decryption permissions. Pioneering the innovative application of PRE in distributed storage systems, Ateniese et al. [54] laid the foundation. FileDES [55] proposes PRE-based encryption methods to protect privacy and against Sybil attacks. He et al. [56] present a framework that integrates PRE and IPFS, demonstrating its efficacy. Kan et al. [57] propose a novel PRE scheme tailored to meet the PoRep scenario, obviating the necessity for the proxy to transform the ciphertext into a new format and consequently reducing computation time.

8. Applications

DSN constitutes a pioneering technological advancement within the realm of storage, with discernible applications spanning a diverse array of industries. In this section, we delve into an exploration of the multifaceted applications of DSNs across various fields.

NFT and Web 3.0. Non-Fungible Token (NFT) represents a manifestation of digital asset ownership within the context of Web 3.0. During the nascent stages of NFT development, storage methodologies for NFT content were predominantly non-decentralized, thereby posing security concerns. In response, Filecoin introduced a dedicated free storage service known as NFT.Storage [58], offering users a secure repository for their NFT content and associated metadata. Notably, OpenSea [59], the world's foremost digital marketplace for NFTs, has integrated Filecoin as its storage tool since 2021. As reported by Grigore et al. [60], the NFT service on Filecoin, as of March 2022, has substantiated the enduring preservation of over 40 million uploads of NFT data, amounting to an aggregate storage capacity exceeding 260 terabytes. Web 3.0 represents a conceptual evolution of the World Wide Web, incorporating tenets such as decentralization, blockchain technologies, and token-based economics [61]. DSNs have evolved in tandem with the Web 3.0 paradigm, serving as a fundamental storage layer within this framework.

Metaverse. The Metaverse, as elucidated by Lee et al. [62], constitutes a virtual environment facilitating online social interaction through digital avatars. To meet the elevated scalability requirements inherent in the metaverse, a robust and decentralized infrastructure is imperative [63,64]. DSN emerges as a critical component in metaverse infrastructure, offering resilience and scalability. An illustrative instance is the Mona project [65], a metaverse initiative grounded in Filecoin, affording users the ability to showcase and trade artistic creations within a virtual realm. Mona ensures that the quality of 3D art models remains uncompromised, obviating the need for concessions to accommodate current dapp bandwidth limitations, thereby preserving the intended aesthetic experience. In a similar vein, Volaverse [66], leveraging Filecoin as its storage layer, pioneers an educational metaverse where users and content creators engage in immersive 3D learning and teaching experiences. Xu et al. [64] propose a trustless metaverse architecture utilizing DSNs to mitigate query latency and furnish privacy guarantees.

Video Streaming. Contemporary internet bandwidth is predominantly consumed by video streaming, and within centralized environments, the storage and streaming of video data incur substantial costs. DSN presents an ideal solution to this predicament.

Huddle [67], a pioneering video conferencing solution rooted in Web 3.0 and blockchain technology, leverages Filecoin for data storage, offering a cost-effective alternative compared to conventional platforms such as Amazon S3 [68]. Livepeer [69], a decentralized live-video streaming infrastructure, optimizes storage and content delivery through Filecoin, incentivizing participants through a mechanism akin to DSNs. Voodfy [70], an early innovative project on the Filecoin platform, functions as a decentralized tool for private video hosting, endowing users with comprehensive control over video content, including the distribution of access rights and strategic decisions on monetization strategies.

Healthcare. Healthcare data, characterized by its scale and privacy considerations, encounters challenges of cost and security when stored on centralized servers. DSNs offer a compelling resolution to these challenges.

Khan et al. [71] propose a scheme utilizing Filecoin for the secure storage of healthcare data, ensuring privacy for sensitive patient information. Adityaa et al. [72] and Subramanian et al. [73] respectively leverage Storj and Filecoin for the storage

of high-resolution medical images and voluminous 3D medical data, effectively reducing the cost of medical data storage. The InterPlanetary Electronic Health Records (IPEHR) [74] application exemplifies a practical decentralized solution, utilizing smart contracts to securely store sensitive medical documents on Filecoin. Data owners maintain control over access rights, enabling authorized stakeholders to securely retrieve medical data from the Filecoin storage.

E-Commerce. The current landscape of commercial data is characterized by concentration within a limited number of entities, conferring exclusive rights to data usage. The decentralized attributes of DSNs offer a potent remedy to this issue, enhancing data security. Smith et al. [75] address the concentration of out-of-home advertising within a limited number of entities through the design of Screencoin. This network, mirroring Filecoin, disrupts advertising monopolies, fostering inclusivity in the advertising commerce domain by allowing broader participation.

9. Challenge and prospective trajectory

The evolutionary phase of DSNs is notably nascent, and while their potential as an enhanced solution for cloud storage in diverse contexts is apparent, they encounter several exigent challenges warranting attention. This section focalizes on the challenges confronted by DSNs, accentuating two pivotal dimensions: scalability and security.

9.1. Scalability challenges

Proof System. The PoS scheme assumes significance within DSNs, ensuring that miners genuinely store the committed data. It constitutes a pivotal mechanism safeguarding data security and availability, albeit at the expense of intricate security encryption designs, entailing high computational overhead.

Illustratively, considering Filecoin, the proof generation phase mandates the computation of a Stacked Depth Robust Graph [76] and a zk-SNARKs proof. The former, a sophisticated data structure, and the latter, a resource-intensive cryptographic tool, collectively incur substantial computational costs. Concurrently, to assure continuous data availability, miners must periodically generate storage proofs, necessitating validation of each sealed sector every 30 min [77]. To qualify as a lotus-miner in Filecoin, a user must configure a system with 256GiB RAM and a GPU featuring no less than 11 GB VRAM [78]. This configuration stringency, while deterring prospective miners, contradicts the inclusive ethos of universal participation. Other DSNs, like Sia and Swarm, simplify their proof processes at the expense of data security. Enhancing the efficiency and security of the proof system represents a critical avenue for future in-depth research.

Cross-chain Interoperability. Cross-chain technology [79], a prominent focus in blockchain research, necessitates analogous attention within DSNs. This technology's primary objective is to facilitate asset, data, and information transfer across diverse blockchain networks, fostering collaborative synergy. While existing DSNs like Filecoin and Storj implement cross-chain services, their interactions are confined to major blockchains such as Bitcoin and Ethereum, excluding other DSNs.

Consider a scenario wherein a new network is subdivided into multiple clusters, each utilizing a distinct DSN for data storage based on application context. For instance, one cluster may use Storj for private data storage, while another opts for Filecoin for publicly accessible data. In such instances, if a Filecoin user wishes to store or retrieve data on Storj, DSNs necessitate designing a smart contract adept at judiciously converting distinct parameters and completing varied forms of storage proof verification.

DSN as CDN. The geographically dispersed distribution of DSN miners, offering caching, scalability, redundancy, reliability, and DDoS mitigation, positions DSNs as prospective Content Delivery Networks (CDNs). However, for DSNs to function comprehensively as CDNs, additional features encompassing load balancing, auditing, and content delivery optimization are imperative.

Filecoin is presently developing two CDN-centric projects, namely Lassie[3] and Saturn.[4] Lassie, a retrieval client for IPFS and Filecoin, integrates diverse data retrieval protocols, including Graphsync supported by Filecoin and Bitswap supported by both IPFS and Filecoin. Saturn, an integration of Lassie, augments functionalities with automatic caching, file exploration, and retrieval acceleration, achieving a 100 ms Average Time to First Byte (TTFB) for IPFS content. Although Storj [7] envisions content delivery in its future plans, implementation remains unrealized due to prioritized developmental tasks.

DSN for CFN. Computing Force Network (CFN) [80], an emergent information infrastructure integrating computing power and network services, aligns seamlessly with the capabilities of DSNs. DSNs, by aggregating storage resources and computing power from independent miners, enhance resource utilization within CFN. Leveraging blockchain ensures credible resource transactions. CFN emerges as a prospective application domain for DSNs, aligning with the objective of democratizing computing power akin to essential utilities.

Multi-Version Control. Presently, DSNs lack the capability to edit uploaded data. To accommodate multi-version files, users resort to sequentially uploading all versions, resulting in significant data redundancy and resource wastage.

Guo et al. [23] propose FileDAG to address multi-version control issues in Filecoin. Implementing file-level deduplication, FileDAG calculates the increment between new and original file versions, storing only the increment in DSN. Despite efficiency gains, challenges persist, notably the linear increase in storage proof calculation costs with an expanding number of file versions. Exploring optimization methods addressing storage and computational costs in the context of multi-version control within DSNs remains a critical area for exploration.

9.2. Security and privacy challenges

Attacks. Introducing satellites in Storj to enhance system efficiency introduces vulnerability, as they operate in a centralized role. A Denial-of-Service (DoS) attack on a satellite, demonstrated by De et al. [81], can render specific Storj functions unusable. This scenario extrapolates to multiple satellites, posing a systemic crash risk. A formal secure analysis of Expected Consensus in Filecoin by Wang et al. [82] highlights the potential n-split attack. Cao et al. [83] explore three temporary block withholding attacks challenging Expected Consensus, indicating a genuine threat to Filecoin.

Parallel to blockchain takeovers, exemplified by the TRON and Steem incident [84], DSNs confront the peril of malicious miners seizing control. A malicious miner with substantial storage resources can manipulate mined blocks, exclusively incorporating its storage proof while discarding proofs from honest miners. As honest miners' proofs remain unverified on the blockchain, they incur penalties, granting rewards to malicious miners. While the success probability of this attack is minimal, its potential ramifications underscore the need to address and mitigate such threats.

Privacy Concerns. Preserving user privacy within DSNs is imperative. However, despite efforts, personal information may be

[3] https://github.com/filecoin-project/lassie

[4] https://saturn.tech/

vulnerable to leakage. In Filecoin's underlying IPFS, peer discovery via the DHT records various information, including unique node identifiers (PeerID) and CIDs. DHT's public visibility facilitates expedited file retrieval but exposes users to potential malicious third-party monitoring of query traffic. Concealing CIDs without compromising data availability and averting DHT-based identification of user IP addresses are crucial challenges.

Illegal Content Monitoring. A DSN operates as an open platform, facilitating the storage of diverse content by any user, including illegal material. This inclusivity extends to illicit content, as evidenced by instances observed within blockchain systems. Matzutt et al. [85] documented the presence of backups for 274 websites containing child pornography within the Bitcoin network. DSNs may inadvertently harbor malware and unlawful content. Furthermore, minor alterations to a file can yield a distinct Content Identifier, thereby obfuscating any correlation between multiple iterations of illicit content. Consequently, the comprehensive removal of such content becomes notably challenging. Employing smart contracts on the blockchain to detect uploaded data poses its own set of challenges. This approach necessitates submitting files as parameters for on-chain computation, yet files stored within DSNs often exhibit considerable size, impeding the efficiency of on-chain computation. Balancing the imperative to identify illegal content with the need for system efficiency poses a significant challenge for DSNs.

Fine-Grained Access Control. Implementing access control mechanisms for files stored within a DSN is pivotal for enhancing data security and privacy [86]. This strategy ensures the safeguarding of sensitive information from unauthorized access, thus preserving the confidentiality and integrity of data dispersed across the distributed network [87]. Presently, the granularity of access control within existing DSNs is markedly coarse, delineated into merely two categories: public access, exemplified by platforms such as Filecoin, where data is universally accessible; and uploader-exclusive access, observed in Storj and Sia, restricting data access solely to the uploader. Nonetheless, clients often seek comprehensive control over their stored files' access rights within a DSN, including the ability to grant or revoke another user's access rights [88]. Consequently, the dynamic requirement of file access rights in DSNs renders the current coarse-grained access control methods inadequate. Thus, developing a fine-grained access control framework for DSNs represents a significant challenge, necessitating innovative approaches to accommodate the complex and evolving requirements of data privacy and security in decentralized storage environments.

10. Conclusion

The DSN represents an innovative paradigm in the field of storage technology. This comprehensive review paper meticulously examines existing DSN projects and scholarly contributions. Through a detailed analysis, it elucidates the operational intricacies of DSN and discusses its effectiveness in ensuring data security and decentralization. Various aspects, including proof of storage, consensus algorithms, and incentive mechanisms, are thoroughly investigated to highlight the multifaceted approach adopted by DSN. The unique features of DSN address the limitations inherent in traditional cloud storage, positioning it as a promising alternative. However, it is essential to recognize that DSN is currently in its early stages of development, facing unresolved challenges and issues. The exploration and resolution of these challenges present compelling opportunities for future research endeavors.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This study was partially supported by the National Key R&D Program of China (2022YFB4501000), the National Natural Science Foundation of China (62232010, 62302266, and U23A20302), Shandong Science Fund for Excellent Young Scholars (2023HWYQ-008), and Shandong Science Fund for Key Fundamental Research Project (ZR2022ZD02).

References

[1] M.R. Palankar, A. Iamnitchi, M. Ripeanu, S. Garfinkel, Amazon S3 for science grids: a viable solution? in: Proceedings of the 2008 International Workshop on Data-Aware Distributed Computing, 2008, pp. 55–64.
[2] S. Challita, F. Zalila, C. Gourdin, P. Merle, A precise model for google cloud platform, in: 2018 IEEE International Conference on Cloud Engineering, IC2E, IEEE, 2018, pp. 177–183.
[3] G. Zhang, M. Ravishankar, Exploring vendor capabilities in the cloud environment: A case study of alibaba cloud computing, Inf. Manage. 56 (3) (2019) 343–355.
[4] X. Cheng, M. Xu, R. Pan, D. Yu, C. Wang, X. Xiao, W. Lyu, Meta computing, IEEE Netw. (2023).
[5] J. Ernstberger, J. Lauinger, F. Elsheimy, L. Zhou, S. Steinhorst, R. Canetti, A. Miller, A. Gervais, D. Song, SoK: Data sovereignty, 2023, Cryptology ePrint Archive.
[6] D. Vorick, L. Champine, Sia: Simple decentralized storage, Retrieved May, vol. 8, 2014, p. 2018.
[7] I. Storj Labs, Storj: A decentralized cloud storage network framework, 2018.
[8] S. Team, SWARM-storage and communication infrastructure for a self-sovereign digital society, 2021.
[9] Protocol Labs, Filecoin: A decentralized storage network, 2017, Retrieved from: https://filecoin.io/filecoin.pdf.
[10] C. Reed, Information in the cloud: ownership, control and accountability, in: Privacy and Legal Issues in Cloud Computing, Edward Elgar Publishing, 2015, pp. 139–159.
[11] AWS, AWS docs, 2023, https://aws.amazon.com/what-is/cloud-storage/.
[12] I. Clarke, O. Sandberg, B. Wiley, T.W. Hong, Freenet: A distributed anonymous information storage and retrieval system, in: Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability Berkeley, CA, USA, July 25–26, 2000 Proceedings, Springer, 2001, pp. 46–66.
[13] A. Muthitacharoen, R. Morris, T.M. Gil, B. Chen, Ivy: A read/write peer-to-peer file system, Oper. Syst. Rev. 36 (SI) (2002) 31–44.
[14] F. Giroire, J. Monteiro, S. Pérennes, P2p storage systems: How much locality can they tolerate? in: 2009 IEEE 34th Conference on Local Computer Networks, IEEE, 2009, pp. 320–323.
[15] C. Williams, P. Huibonhoa, J. Holliday, A. Hospodor, T. Schwarz, Redundancy management for P2P storage, in: Seventh IEEE International Symposium on Cluster Computing and the Grid, CCGrid'07, IEEE, 2007, pp. 15–22.
[16] I. Osipkov, P. Wang, N. Hopper, Robust accounting in decentralized P2P storage systems, in: 26th IEEE International Conference on Distributed Computing Systems, ICDCS'06, IEEE, 2006, p. 14.
[17] D. Trautwein, A. Raman, G. Tyson, I. Castro, W. Scott, M. Schubotz, B. Gipp, Y. Psaras, Design and evaluation of IPFS: a storage layer for the decentralized web, in: Proceedings of the ACM SIGCOMM 2022 Conference, 2022, pp. 739–752.
[18] P. Maymounkov, D. Mazieres, Kademlia: A peer-to-peer information system based on the xor metric, in: International Workshop on Peer-To-Peer Systems, Springer, 2002, pp. 53–65.
[19] J. Benet, Ipfs-content addressed, versioned, p2p file system, 2014, arXiv preprint arXiv:1407.3561.
[20] E. Daniel, F. Tschorsch, IPFS and friends: A qualitative comparison of next generation peer-to-peer data networks, IEEE Commun. Surv. Tutor. 24 (1) (2022) 31–52.
[21] M. Song, J. Han, H. Eom, Y. Son, IPFSz: An efficient data compression scheme in InterPlanetary File System, IEEE Access 10 (2022) 122601–122611.
[22] J. Sun, X. Yao, S. Wang, Y. Wu, Blockchain-based secure storage and access scheme for electronic medical records in IPFS, IEEE Access 8 (2020) 59389–59401.
[23] H. Guo, M. Xu, J. Zhang, C. Liu, D. Yu, S. Dustdar, X. Cheng, FileDAG: A multi-version decentralized storage network built on DAG-based blockchain, IEEE Trans. Comput. (2023).
[24] I. Keidar, E. Kokoris-Kogias, O. Naor, A. Spiegelman, All you need is dag, in: Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing, 2021, pp. 165–175.

[25] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, D. Song, Provable data possession at untrusted stores, in: Proceedings of the 14th ACM Conference on Computer and Communications Security, 2007, pp. 598–609.

[26] A. Juels, B.S. Kaliski Jr., PORs: Proofs of retrievability for large files, in: Proceedings of the 14th ACM Conference on Computer and Communications Security, 2007, pp. 584–597.

[27] A.G. Dimakis, V. Prabhakaran, K. Ramchandran, Decentralized erasure codes for distributed networked storage, IEEE Trans. Inform. Theory 52 (6) (2006) 2809–2816.

[28] Q. Zheng, S. Xu, Secure and efficient proof of storage with deduplication, in: Proceedings of the Second ACM Conference on Data and Application Security and Privacy, 2012, pp. 1–12.

[29] Storj, Storj docs, 2023, https://docs.storj.io/learn/concepts/satellite.

[30] J. Benet, D. Dalrymple, N. Greco, Proof of replication, Protocol Labs, July, vol. 27, 2017, p. 20.

[31] H. Qi, Y. Cheng, M. Xu, D. Yu, H. Wang, W. Lyu, Split: A hash-based memory optimization method for zero-knowledge succinct non-interactive argument of knowledge (zk-SNARK), IEEE Trans. Comput. (2023).

[32] I. Damgård, C. Ganesh, C. Orlandi, Proofs of replicated storage without timing assumptions, in: Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part I 39, Springer, 2019, pp. 355–380.

[33] E. Cecchetti, B. Fisch, I. Miers, A. Juels, PIEs: Public incompressible encodings for decentralized storage, in: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, 2019, pp. 1351–1367.

[34] H. Yuan, X. Chen, G. Xu, J. Ning, J.K. Liu, R.H. Deng, Efficient and verifiable proof of replication with fast fault localization, in: IEEE INFOCOM 2021-IEEE Conference on Computer Communications, IEEE, 2021, pp. 1–10.

[35] C. Zhang, X. Li, M.H. Au, ePoSt: Practical and client-friendly proof of storage-time, IEEE Trans. Inf. Forensics Secur. 18 (2023) 1052–1063.

[36] T. Moran, I. Orlov, Simple proofs of space-time and rational proofs of storage, in: Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part I 39, Springer, 2019, pp. 381–409.

[37] G. Ateniese, L. Chen, M. Etemad, Q. Tang, Proof of storage-time: Efficiently checking continuous data availability, 2020, Cryptology ePrint Archive.

[38] R. Rabaninejad, B. Abdolmaleki, G. Malavolta, A. Michalas, A. Nabizadeh, stoRNA: Stateless transparent proofs of storage-time, 2023, Cryptology ePrint Archive.

[39] R. Rabaninejad, B. Liu, A. Michalas, PoRt: Non-interactive continuous availability proof of replicated storage, in: Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing, 2023, pp. 270–279.

[40] M. Xu, Y. Guo, C. Liu, Q. Hu, D. Yu, Z. Xiong, D. Niyato, X. Cheng, Exploring blockchain technology through a modular lens: A survey, 2023, arXiv preprint arXiv:2304.08283.

[41] S. King, S. Nadal, Ppcoin: Peer-to-peer crypto-currency with proof-of-stake, 2012, self-published paper, August, vol. 19, no. 1.

[42] F. Victor, B.K. Lüders, Measuring ethereum-based erc20 token networks, in: Financial Cryptography and Data Security: 23rd International Conference, FC 2019, Frigate Bay, St. Kitts and Nevis, February 18–22, 2019, Revised Selected Papers 23, Springer, 2019, pp. 113–129.

[43] F. Saleh, Blockchain without waste: Proof-of-stake, Rev. Financ. Stud. 34 (3) (2021) 1156–1190.

[44] Filecoin, Filecoin docs, 2023, https://docs.filecoin.io/.

[45] Filecoin, Filecoin spec, 2023, https://spec.filecoin.io/.

[46] V.H. Lakhani, L. Jehl, R. Hendriksen, V. Estrada-Galinanes, Fair incentivization of bandwidth sharing in decentralized storage networks, in: 2022 IEEE 42nd International Conference on Distributed Computing Systems Workshops, ICDCSW, IEEE, 2022, pp. 39–44.

[47] V.H. Lakhani, L. Jehl, G. Ishmaev, V. Estrada-Galiñanes, Tit-for-token: fair rewards for moving data in decentralized storage networks, 2023, arXiv preprint arXiv:2307.02231.

[48] L. Rizzo, Effective erasure codes for reliable computer communication protocols, ACM SIGCOMM Comput. Commun. Rev. 27 (2) (1997) 24–36.

[49] H. Guo, M. Xu, J. Zhang, C. Liu, R. Ranjan, D. Yu, X. Cheng, BFT-DSN: A Byzantine fault tolerant decentralized storage network, IEEE Trans. Comput. (2024) 1–13.

[50] J.R. Douceur, The sybil attack, in: International Workshop on Peer-To-Peer Systems, Springer, 2002, pp. 251–260.

[51] B. Prünster, A. Marsalek, T. Zefferer, Total eclipse of the heart–disrupting the {InterPlanetary} file system, in: 31st USENIX Security Symposium (USENIX Security 22), 2022, pp. 3735–3752.

[52] S. Wang, Y. Zhang, Y. Zhang, A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems, Ieee Access 6 (2018) 38437–38450.

[53] M. Steichen, B. Fiz, R. Norvill, W. Shbair, R. State, Blockchain-based, decentralized access control for IPFS, in: 2018 Ieee International Conference on Internet of Things (IThings) and Ieee Green Computing and Communications (GreenCom) and Ieee Cyber, Physical and Social Computing (CPSCom) and Ieee Smart Data (SmartData), IEEE, 2018, pp. 1499–1506.

[54] G. Ateniese, K. Fu, M. Green, S. Hohenberger, Improved proxy re-encryption schemes with applications to secure distributed storage, ACM Trans. Inf. Syst. Secur. 9 (1) (2006) 1–30.

[55] M. Xu, J. Zhang, H. Guo, X. Cheng, D. Yu, Q. Hu, Y. Li, Y. Wu, FileDES: A secure, scalable and succinct decentralized encrypted storage network, 2024, Cryptology ePrint Archive, Paper 2024/182, [Online]. Available: https://eprint.iacr.org/2024/182.

[56] J. He, D. Zheng, R. Guo, Y. Chen, K. Li, X. Tao, Efficient identity-based proxy re-encryption scheme in blockchain-assisted decentralized storage system, Int. J. Netw. Secur. 23 (5) (2021) 776–790.

[57] J. Kan, J. Zhang, D. Liu, X. Huang, Proxy re-encryption scheme for decentralized storage networks, Appl. Sci. 12 (9) (2022) 4260.

[58] Protocol Labs, Nft.storage docs, 2023, https://nft.storage/docs/.

[59] OpenSea, OpenSea, 2023, https://opensea.io/.

[60] M. Grigore, S. Kassab, Filecoin has it: An ecosystem overview, 2022, https://messari.io/report/filecoin-has-it-an-ecosystem-overview.

[61] M. Fenwick, P. Jurcys, The contested meaning of Web3 and why it matters for (IP) lawyers, 2022, Available at SSRN 4017790.

[62] L.-H. Lee, T. Braud, P. Zhou, L. Wang, D. Xu, Z. Lin, A. Kumar, C. Bermejo, P. Hui, All one needs to know about metaverse: A complete survey on technological singularity, virtual ecosystem, and research agenda, 2021, arXiv preprint arXiv:2110.05352.

[63] Y. Cheng, Y. Guo, M. Xu, Q. Hu, D. Yu, X. Cheng, An adaptive and modular blockchain enabled architecture for a decentralized metaverse, IEEE J. Sel. Areas Commun. (2023) 1.

[64] M. Xu, Y. Guo, Q. Hu, Z. Xiong, D. Yu, X. Cheng, A trustless architecture of blockchain-enabled metaverse, High-Confidence Comput. 3 (1) (2023) 100088.

[65] Mona, Mona docs, 2023, https://docs.monaverse.com/.

[66] Volaverse, Volaverse, 2023, https://www.volaverse.com/.

[67] Huddle01, Huddle01, 2023, https://huddle01.com/.

[68] Filecoin, Filecoin for media, video, gaming, and more, 2021, https://filecoin.io/blog/posts/filecoin-for-media-video-gaming-and-more/.

[69] D. Petkanics, E. Tang, Livepeer whitepaper, 2023, https://github.com/livepeer/wiki/blob/master/WHITEPAPER.md.

[70] Voodfy, Voodfy, 2021, https://github.com/Voodfy.

[71] A.A. Khan, A.A. Wagan, A.A. Laghari, A.R. Gilal, I.A. Aziz, B.A. Talpur, BIoMT: A state-of-the-art consortium serverless network architecture for healthcare system using blockchain smart contracts, IEEE Access 10 (2022) 78887–78898.

[72] G. Adityaa, V. Lavanya, A decentralized storage system for 3D medical data with dynamic AES and AES-GCM encryption, in: Recent Developments in Electronics and Communication Systems: Proceedings of the First International Conference on Recent Developments in Electronics and Communication Systems, Vol. 32, RDECS-2022, IOS Press, 2023, p. 269.

[73] H. Subramanian, S. Subramanian, Improving diagnosis through digital pathology: Proof-of-concept implementation using smart contracts and decentralized file storage, J. Med. Internet Res. 24 (3) (2022) e34207.

[74] B. Supernova, IPEHR, 2023, https://github.com/bsn-si/IPEHR-gateway.

[75] M. Smith, A. Castro, M. Rahouti, M. Ayyash, L. Santana, ScreenCoin: A blockchain-enabled decentralized Ad network, in: 2022 IEEE International Conference on Omni-Layer Intelligent Systems, COINS, IEEE, 2022, pp. 1–6.

[76] J. Alwen, J. Blocki, K. Pietrzak, Depth-robust graphs and their cumulative memory complexity, in: Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 2017, pp. 3–32.

[77] B. Guidi, A. Michienzi, L. Ricci, Evaluating the decentralisation of filecoin, in: Proceedings of the 3rd International Workshop on Distributed Infrastructure for the Common Good, 2022, pp. 13–18.

[78] Protocol Labs, Lotus, 2023, https://lotus.filecoin.io/storage-providers/get-started/hardware-requirements/.

[79] Y. Guo, M. Xu, D. Yu, Y. Yu, R. Ranjan, X. Cheng, Cross-channel: Scalable off-chain channels supporting fair and atomic cross-chain operations, IEEE Trans. Comput. (2023).

[80] X. Shi, Q. Li, D. Wang, L. Lu, Mobile Computing Force Network (MCFN): Computing and network convergence supporting integrated communication service, in: 2022 International Conference on Service Science, ICSS, 2022, pp. 131–136.

[81] S. de Figueiredo, A. Madhusudan, V. Reniers, S. Nikova, B. Preneel, Exploring the storj network: A security analysis, in: Proceedings of the 36th Annual ACM Symposium on Applied Computing, 2021, pp. 257–264.

[82] X. Wang, S. Azouvi, M. Vukolić, Security analysis of filecoin's expected consensus in the Byzantine vs honest model, 2023, arXiv preprint arXiv:2308.06955.

[83] T. Cao, X. Li, Temporary block withholding attacks on filecoin's expected consensus, in: Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses, 2023, pp. 109–122.
[84] C. Li, B. Palanisamy, R. Xu, L. Duan, J. Liu, W. Wang, How hard is takeover in DPoS blockchains? Understanding the security of coin-based voting governance, in: Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, 2023, pp. 150–164.
[85] R. Matzutt, J. Hiller, M. Henze, J.H. Ziegeldorf, D. Müllmann, O. Hohlfeld, K. Wehrle, A quantitative analysis of the impact of arbitrary blockchain content on bitcoin, in: Financial Cryptography and Data Security: 22nd International Conference, FC 2018, Nieuwpoort, CuraÇAo, February 26–March 2, 2018, Revised Selected Papers 22, Springer, 2018, pp. 420–438.
[86] S. Namane, I. Ben Dhaou, Blockchain-based access control techniques for iot applications, Electronics 11 (14) (2022) 2225.
[87] C. Liu, M. Xu, H. Guo, X. Cheng, Y. Xiao, D. Yu, B. Gong, A. Yerukhimovich, S. Wang, W. Lyu, TBAC: A Tokoin-based accountable access control scheme for the internet of things, IEEE Trans. Mob. Comput. (2023) 1–16.
[88] A. Chatterjee, Y. Pitroda, M. Parmar, Dynamic role-based access control for decentralized applications, in: Blockchain–ICBC 2020: Third International Conference, Held As Part of the Services Conference Federation, SCF 2020, Honolulu, HI, USA, September 18-20, 2020, Proceedings 3, Springer, 2020, pp. 185–197.