Efficient Decentralized Data Storage Based on Public Blockchain and IPFS

Morteza Alizadeh*, Karl Andersson*, and Olov Schelén*
* *Department of Computer Science, Electrical and Space Engineering*
Luleå University of Technology
SE-93187 Skellefteå, Sweden
Email: {morteza.alizadeh, karl.andersson, olov.schelen}@ltu.se

2020 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE) | 978-1-6654-1974-1/20/$31.00 ©2020 IEEE | DOI: 10.1109/CSDE50874.2020.9411599

***Abstract*—Blockchain technology has enabled the keeping of a decentralized, tamper-proof, immutable, and ordered ledger of transactional events. Efforts to leverage such a ledger may be challenging when data storage requirements exceed most blockchain protocols' current capacities. Storing large amounts of decentralized data while maintaining system efficiency is the challenge that we target. This paper proposes using the IPFS distributed hash table (DHT) technology to store information immutably and in a decentralized manner to mitigate the high cost of storage. A storage system involving blockchain and other storage systems in concert should be based on immutable data and allow removal of data from malicious users in the DHT. Efficiency is improved by decreasing the overall processing time in the blockchain with the help of DHT technology and introducing an agreement service that communicate with the blockchain via a RESTful API. We demonstrate the applicability of the proposed method and conclude that the combination of IPFS and blockchain provides efficient cryptographic storage, immutable history and overall better efficiency in a decentralized manner.**

***Index Terms*—Public blockchain, smart contract, distributed hash table, immutability.**

I. Introduction

Decentralization is the process of distributing applications, functions, power, people, or things away from a central location or authority. If a system is decentralized, it means that it is not controlled, owned, or managed by a single person or authority. The major application domains are home and office automation, transportation, environmental monitoring, healthcare, entertainment, education, and security [1]. Typical requirements on decentralized applications include short response times, enough storage capacity, and avoiding a single point of failure [2].

Blockchain is a decentralized system with a good level of security for transactions. It provides an environment for distributed applications to process and store information in an immutable and secure system [1]–[9].

Blockchain stores a cryptographic signature of data or records of data. Moreover, it has several functions inside to address how to secure the transactional data and prevent malicious cyber-attacks. Typically, blockchain should store transactions' data in its network.

Keeping stored data immutable is one of the decentralized storage system's objectives [3]. The general meaning of immutable data is keeping data in computers, electronic systems, and networks unchangeable. Blockchain is typically used because it provides a solution for keeping an immutable global event or transaction ordering. It involves protecting storage resources and information from accidental or deliberate damage or destruction and unauthorized or malicious users accessing open and public systems.

The validation process is time-consuming process [10]. Data storage limitation is another issue in the blockchain.

The InterPlanetary File System (IPFS) is a decentralized file system that uses Distributed Hash Table (DHT) technology [10], [11]. It helps to store any size of data decentralized without storage limitations. This paper's main contribution is to show how IPFS can help blockchain to be more efficient with shorter response times and overcome storage limitations. This paper demonstrates the applicability of our proposed solution in two scenarios. These scenarios describe handling of agreements between two parties by the help of IPFS and blockchain technology.

An agreement is an electronic document that includes negotiated information among parties like assets and signatures stored in centralized or decentralized storage systems. Web services and IPFS can cover agreement storage issues decentralized. Smart contract, as the fundamental part of Ethereum blockchain technology, can communicate with these services and IPFS by having a RESTful API connection link. Accordingly, minimizing the number of referrals to the blockchain network is essential for decentralized systems. Pushing the agreements to a combination of blockchain and IPFS in one request is the main target of this paper. We choose to combine IPFS and blockchain to have an efficient and immutable system. In other words, connected parties to the system should send one agreement to the blockchain together after uploading data to IPFS instead of sending many separated requests.

Scenario (2) in section IV describes mitigation of storage limitations and long response times, while minimizing the number of referrals to the blockchain with the help of a special agreement service as a junction between the IPFS and

blockchain.

This paper has been organized in the following manner. Section II shows recent research within blockchain and IPFS. Section III discusses the blockchain definition, how it can be beneficial in distributed networks, and its purposes. This section also introduces DHT technology and common storage problems in a decentralized system. Section IV illustrates possible solutions and shows scenarios relative to the IPFS and blockchain, while section V evaluates our proposed solution. Section VI concludes the paper and outlines future work.

II. Related work

The blockchain system is a decentralized system that stores transactional records. Common use cases are the usage of blockchains to enable a tamper-proof log of events [12] and the distribution of data storage [13]. Zhijie et al. [14] explained how to exchange data of distributed events in supply chains via the blockchain. Haque et al. [15] used the IPFS as their data-sharing infrastructure and blockchain for transferring data in a peer-to-peer network to transport pre-trained deep learning models to others. Shah et al. [16] combined blockchain with cloud technology to mitigate data security, privacy, availability, and resource utilization. Among other decentralized storage systems research [11], [17], [18], the common view is that blockchain can be used for data sharing purposes. Data storage with the help of blockchain and the IPFS is presented in their work on establishing a cloud and blockchain-based infrastructure for data exchange and leverages the blockchain to provide an access controlled IPFS. Tenorio et al. [19] proposed solutions for data access, data provenance, and data discovery by suggesting a framework for the design of decentralized systems and provided a set of guidelines to decide when and where blockchain technology may be required, or when other technologies, such as the IPFS, are sufficient.

The combination of blockchain and IPFS is a new technique in decentralized systems to cover weaknesses like storage limitation and time-consuming issues in blockchain and immutably recorded transactions in the IPFS. Communicating and interacting beyond blockchain to other decentralized systems is the innovation of this paper compared with others. Solving storage limitations in blockchain and mitigating the problem with long response times are two results of this paper.

III. Blockchain and DHT Terminology

Public, private, and consortium blockchain are three types of blockchain in decentralized systems [3]. The cryptocurrency concept is common in the public blockchain and main focus of this paper is about solving public blockchains' existed issues.

A. Blockchain and Smart Contracts

Blockchain stores transactions' details in a distributed fashion. This distributed technology provides immutability and trust [5] and is beneficial for record keeping, digital notary, and smart contracts. It has been initially used for digital currency [6] and secure distributed transaction storage systems. Bitcoin is a famous case of cryptocurrency. It operates decentralized digital currency by removing central management [7]. Increasing trustability is a result of using cryptography. A High level view of a transaction in Ethereum is illustrated in Figure 1. Buyers and dealers want to communicate with public blockchain to have a secure transaction in the decentralized network.

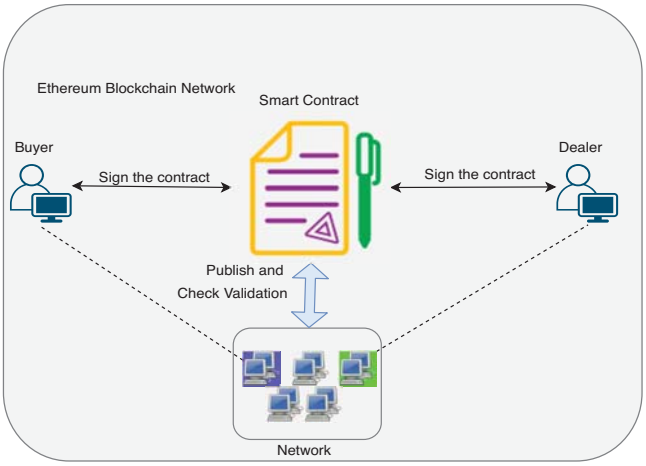

Fig. 1. Schema of public blockchain's transaction

The blockchain occupant is a set of data packages called blocks. A set of blocks represents a complete ledger of the transaction history. Each block contains multiple records such as timestamps and a hash code, and there is a record in the block assigned to the previous block. The previous block is known as a parent block. It is possible to return to the first block or the genesis block by following the parent blocks. This theory secures the integrity of the whole blockchain. The block hash value is unique for each data. The blockchain's block will be added when most of the parties in the network agree by a consensus mechanism [8]. The data will be published across several nodes among networked devices. All nodes replicate and preserve the same copy of the ledger after their consensus operation is finished. The agreed upon version of the ledger by other members of the network is stored on each node separately and causes trust. This is the reason that stored information in the blockchain is unchangeable. It is crucial to pay for operations and transactions with virtual money or credits. These credits motivate others to prove the correctness of the information in the network. Further, the network participants are motivated to compete in a competition to catch more credits.

B. Storage Limitation and Time Problems

Storage and time problems are considered two problems of some decentralized storage systems such as Ethereum. Therefore, decentralized storage systems should be designed well to guarantee these aspects for network members. Although, blockchains as decentralized technologies help to solve data immutably, and they have storage and time problems.

In this paper, the problems are divided into two issues; the first issue is about storage capacity problem, and the next issue concerns the time-consuming problem.

1) Storage Problems of the Blockchain System: One of the common problems of large, decentralized networks is data storage when the number of referrals for storage in the system increases. This issue is a frequent problem in most networks. A public blockchain is regarded as a solution to keep information immutable in a decentralized system. Nevertheless, a lack of memory to store everything decentralized with different data sizes is known as a blockchain issue.

2) Time Consuming Processes in Blockchain: Blockchain technology has many functions with high time consumption. Every approved transaction requires peer-to-peer verification, which can become time-consuming with the number of involved blocks. The validation process, consensus algorithm, adding a new block, adding transaction information to the block, distributing replica to all users, and waiting for other users to come and accept are time-consuming blockchain functions.

C. Ethereum

Ethereum is one of the largest cryptocurrencies. Ether is a cryptocurrency generated by Ethereum miners as a reward for computations performed well in the blockchain. It is a decentralized system and works by running smart contract functionality. Ethereum has a decentralized virtual machine (EVM). The EVM can execute scripts by having a public network. The EVM uses a gas parameter as an internal transaction pricing mechanism. The consensus in the EVM is based on proof of work (PoW) initially. PoW is the consensus algorithm used in Ethereum which is used in cryptocurrencies. Parties in the system compete against each other to complete transactions on the network and get rewarded. Proof of stake (PoS) is an alternative to PoW. The producer of the block will be selected via different combinations of random selection and wealth or age. Ethereum developers has been trying to switch from PoW to PoS. PoS protects the system from having selfish mining attacks [1].

D. Distributed Systems and Hash Tables

A hashed data is an output of a function that converts one value to another value with different methods. Hash is the string that is the output of a hashing algorithm such as MD5 (Message Digest 5) or SHA (secure hash algorithm) [20]. It is used for several different areas such as cryptography, one-way compression, and data indexing. A hash function can be used to generate a value that can only be decoded by looking up the value from a hash table. The table may be an array, database, or other data structure. A good cryptographic hash function is one-way and noninvertible.

A hash table is a data structure that fulfills the association of abstract data type. A hash table uses a hash function to figure an index or hash code. It can map keys to values. It pushes hash codes into an array of buckets or slots. It is easy to look up or find values in it. Hash tables are more efficient than search trees or any other table lookup structure in many cases. The lookup means that the hashed value's address and the resulting hash refers to the corresponding stored value.

A DHT is a distributed form of hash table. Distributed systems are fully connected systems, where parties are located in different places and there is no significant difference between parties. Therefore, all parties can keep a copy of most recently updated hash table. The DHT has a lookup service similar to the hash table. It uses a hash table where (key, value) pairs are stored in it. Any participating node can recover the hashed value linked with a given key. The main advantages of the DHT is that all nodes can be added/removed at a minimum time just by re-distributing the keys [21]. Hash code helps identify during identity examination and can run as indexing system. It can develop blockchains to be faster [9].

E. IPFS

The IPFS is a peer-to-peer distributed file system that uses a DHT to track who has what data. Additionally, it is a new model of sharing file distributed. The IPFS has a web application that makes easy for users to work with it, as shown in Figure 2. The IPFS uses hash tables to store a data package. IPFS nodes can provide blocks of data. The IPFS uses Kademlia to learn which nodes have what data. Kademlia is a DHT for decentralized peer-to-peer computer networks designed by Petar Maymounkov and David Mazières in 2002 [21]. A unique hash is a result of saving data without worrying about data size in the IPFS. The IPFS can store the hash and then parties can use the hash to retrieve the data. When data is ready to add on the IPFS network, the data will split into many small chunks. The chunk is identified with its own hash. Then, the chunks will be distributed to various nodes on the network which have their hash closest to peer_Id. Once the user requests to retrieve a chunk, the retrieve request traverses to nodes where the hash exists there by using the DHT. All the chunks are simply combined to show the main object after visiting all the existed chunks. However, the distributed part of DHT means that the entire table is spread across different locations.

Fig. 2. IPFS peers

IV. Blockchain, Smart Contract Solutions and Scenarios

There are two scenarios in this section that illustrate how parties can interact with blockchain and IPFS. The first scenario shows a system without storage limitation issue, while the second scenario is an improvement of the first scenario to handle and manage the time.

There are two different concepts explained in this paper: 1) an agreement and 2) a smart contract.

An agreement is digitally signed with an initiator and responder that include contractual information, addresses, and references to previously signed agreements. A decentralized system can store an agreement decentralized. It is a signed or written document between actors after they accept all terms of written inside. For the sake of simplicity, in this paper we assume only two parties that agree signing an agreement (not multi-party agreements).

A smart contract is a set of lines of computer code that is stored on a blockchain and is automatically performed when any party in the network wants to execute a smart contract. In this paper, we use the smart contract to log agreements in a serialized and immutable manner. We also control and manage relevant events and functions according to the terms other smart contracts.

The process begins with a user request to the blockchain. Then, network members replied that blockchain stores transaction information and data in a distributed shared database after validation and acceptance. Immutable history of transactions prevent malicious users to perform destructive actions in the system. It is clear that blockchain improves the whole system capability to prevent most security threats by hashing, chaining, and distributing the ledger definition. The security problem of transactions or events is already improved by using public blockchain as a decentralized system, because of immutable history of data. Nobody can change it because everyone carries a copy of that in their memory.

The blockchain provides trust and immutability without central authority. There are reasons to use blockchain and have an efficient system.

A. Data Storage Limitation Solution

The IPFS is solution for data storage limitation problem. It helps blockchain to solve storage limitation problem. Decentralized and immutable data will be kept in the IPFS. Blockchain holds the hash address of data location as a field on the transaction data record. Data can be stored in the IPFS, and information about transactions includes addresses and signatures, will be kept in the blockchain. IPFS can solve data storage limitations in blockchain by considering the mentioned approach.

As a summary of the solution section, public blockchain helps whole systems have an immutable history of data, the IPFS helps blockchain to store data immutably without limitations, and smart contracts perform as a type of bridge that connects the IPFS and blockchain. Moreover, this combination can be useful to prevent malicious users to commit their activities.

B. Less Time-Consuming Solution

Smart contracts can handle exchange money, investments, shares, or many use cases in e-commerce. The asset should be paid to another party or forward it to the other account wallet after agreement. The data block of an agreement can be stored on the blockchain with the help of a smart contract in the system. All parties who want to follow these smart contract roles must sign and accept the smart contract's written terms. Smart contracts have the capability to communicate with other services with the aid of RESTful API [22] communication.

C. Scenarios and Methods

We express two scenarios using public blockchain and the IPFS in the Solidity language. The first scenario shows a simple form of communication with blockchain and stores information in the IPFS in parallel. The second scenario is the proposed method in this paper. The presence of a single-call blockchain instead of a multiple call is the reason that we select this scenario.

Smart contracts should be designed as a base of needs, architectures, and examine how to send, receive, and store data.

1) Simple Scenario: In the first scenario (scenario 1), as shown in Figure 3, parties want to communicate through the network. The data object looks similar to a JSON file covering all information that the smart contract needs to execute. Meanwhile, DHT technology helps the system to store records of data. They want to sign an agreement after negotiation. The parties try to create a record of data in the IPFS and announce it on the blockchain. After the announcement and publishing information as a block, other network members can see this block. If they agree with the content, they will decide to accept and sign it, either reject and ignore it. When all parties shows their agreement, smart contract will perform their requests, then transaction will be published into the network after being validated by the system. It will be registered in the system. The transaction is a process that shows all the above steps by storing the information in the blockchain. This transaction stores all information includes two side signatures, addresses, and smart contract terms. This information will follow up, and even this will leverage the network to prevent some security threats. This scenario helps to mitigate storage limitation problem.

2) Blockchain in One Go Scenario: In the second scenario (scenario 2), two actors want to communicate through the network, as shown in Figure 4. They sign an agreement after negotiation. Thereafter, the agreement including signatures from both parties is pushed to IPFS. Then the hash of the agreement is entered to the blockchain. Blockchain stores the has of agreement, parties addresses, timestamp and all information in the agreement service. The agreement negotiation service is another component outside the scope of this paper. A smart contract is designed to connect to other services

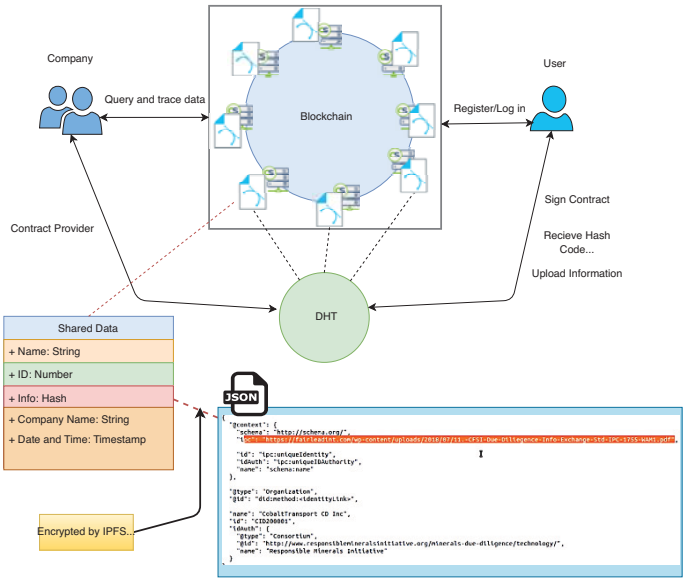

Fig. 3. Simple scenario

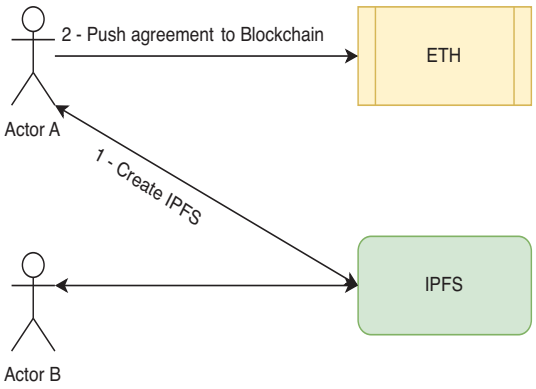

Fig. 4. Blockchain in One Go scenario

and call query from those services. These services can be a data storage service, key server, and any other services. The agreement is a service that smart contracts communicate with and read data from and write data to. The standard library that enables smart contracts to communicate with the other servers is Provable, which communicates using RESTful APIs. In the next step, the agreement between the two actors will be published and distributed as a block of information. All system users can see this information. When users upload an agreement to the blockchain, the transaction will be published after being validated by the system. Therefore, all these steps information as a transaction will be stored in the blockchain. This transaction stores all information including two side signatures, identities and addresses. This method helps the system to prevent attacks. This scenario has one significant difference with first scenario, namely the blockchain's functions being called only once instead of multiple times. This scenario as an efficient solution helps to remove referral times for blockchain functions in compare with first scenario. Then, sending many events to the blockchain will be less than first scenario. This scenario helps to improve time problem.

V. Evaluation

In this section, blockchain in the One Go architecture is discussed as the main approach in four steps: 1) IPFS, 2) agreement, 3) smart contract, and 4) storing data in blockchain.

Figure 5 illustrates a sequence diagram that shows how this approach works and these four steps.

There are two ways to use the IPFS, namely, the IPFS desktop version or the command line version. The desktop app has a visual effect and an easy interface for adding, pinning, and sharing files. IPFS Companion is a browser extension that makes the command line slightly easier to work with.

Each IPFS step must store the big size of data decentralized that can be a JSON file structure. The output of this section is a hash code of data as an address of data source. It is easy to access the data by knowing this hashed address and gateways addresses. It is easy to recover the data in the IPFS because the file is stored in many places. It will not have any problem after losing one of them.

Agreement is the next section. Both sides should read and accept the agreement before sending it to the blockchain. This agreement includes both sides addresses, hashed values by the IPFS, and some more information collected in a JSON file. Then, the accepted registered agreement will be forwarded to the blockchain. Agreement Service combines all information that comes from two parties, then the blockchain can access that by the help of RESTful API requests and less query time.

Provable (Oraclize) is a famous library, or an external smart contract, that includes many functions to empower Solidity smart contract to communicate with other services with RESTful API.

Ethereum is one popular public blockchain, and Remix IDE is a famous editor for compiling and running the smart contract written by Solidity language in Ethereum. Remix has an online version and offline. Metamask is an Ethereum wallet that includes accounts, addresses, and credit (ETH) is necessary to sign up then connect it to the blockchain. We used the Ethereum Kovan test network for testing smart contracts in a public blockchain.

After the validation process, blockchain can add new transaction data in its immutable historical data.

Several significant parameters are essential in the systems: 1) time, 2) storage, and 3) performance. The performance defines based on time and data storage:

$$Time_{Blockchain} = 2 \times (T_{request} + T_{response} + T_{BC}) \quad (1)$$

$$Time_{S1} = 2 \times (T_{request} + T_{response} + T_{BC}) + T_{IPFS} \quad (2)$$

$$Time_{S2} = T_{request} + T_{response} + T_{BC} + T_{IPFS} + T_{SC} \quad (3)$$

$$Time_{Blockchain} = 2 \times T_{total} \quad (4)$$

$$Time_{S1} = 2 \times T_{total} + T_{IPFS} \quad (5)$$

$$Time_{S2} = T_{total} + T_{IPFS} + T_{SC} \quad (6)$$

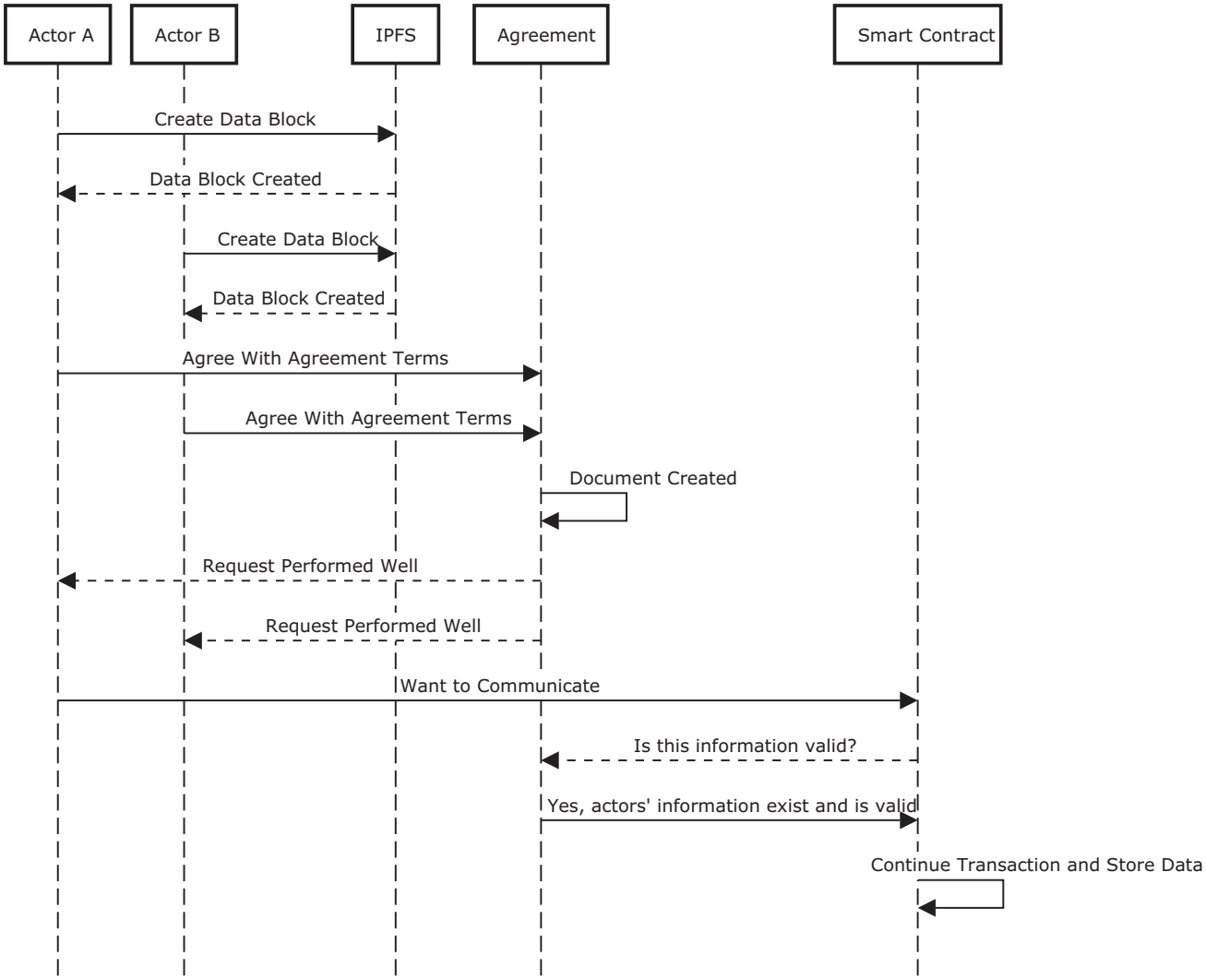

Fig. 5. Proposed transaction flow

$$T_{total} > T_{SC} \Rightarrow Time_{S1} > Time_{S2} \quad (7)$$

A system has high performance when it can store any size of data within the shortest possible time. There are comparisons in Table I that consider the order of the three scenarios. Scenarios 1 and 2 are compered with a scenario without a DHT that has a problem with large amounts of data. Performance is defined by considering storage capacity and time consumed. These parameters' value is divided into three levels (low, medium, high). The scenario 2 performance is the best compared to that of the others because it is less time-consuming and has no storage limitation problem. Both scenarios 1 and 2 contain many functions that check for validating and distributing the ledgers in the blockchain. Thus, these systems are slower, as shown in Eqs. (1), (2), and (3). T_{BC} is time for performing blockchain functions, T_{IPFS} is time for IPFS uploading, $T_{request}$, $T_{response}$ are times for user communication with system, and T_{SC} is a time for calling agreement service by smart contract that is related to service and time to execution. This time, scenario 1 is slower than scenario 2 where a party performs the connection manually according to Eq. (7). $T_{request}$, $T_{response}$, and T_{BC} are same arguments in all scenarios. Then we considered all of them T_{total} as a constant argument in Eqs. (4), (5), and (6). T_{IPFS} is a significant argument. There are three main functions (add, ping, connect to daemon) to upload a file to IPFS. We repeat uploading files of sizes 443 KB and 55.47 MB to IPFS 100 times. The results are illustrated in Figure 6 and Figure 7, where variances for the three functions are indicated in each caption. We use a MacBook Pro (15-inch, 2018) laptop with the following configuration: macOS Big Sur Version 11.0.1, Processor 2,2 GHz 6-Core Intel Core i7, Memory 16 GB 2400 MHz DDR4, average download data rate 95 Mbps, average upload data rate 91 Mbps, and IPFS version 0.7.0.

Connection time is a fluctuating parameter that is depending on speed of real time connection. Adding times are higher

as shown in Figure 7. Larger size of data is the reason of that. Pinning times are almost the same. Scenarios 1 and 2 have higher uploading data volumes to IPFS compared with a system without IPFS storage technology (blockchain). Time consumption is higher in scenario 1 because of multiple referral times to the blockchain. Scenario 2 has the best performance because of the single time that it sends a request to the blockchain and its lack of limitation for storing as shown in Table I.

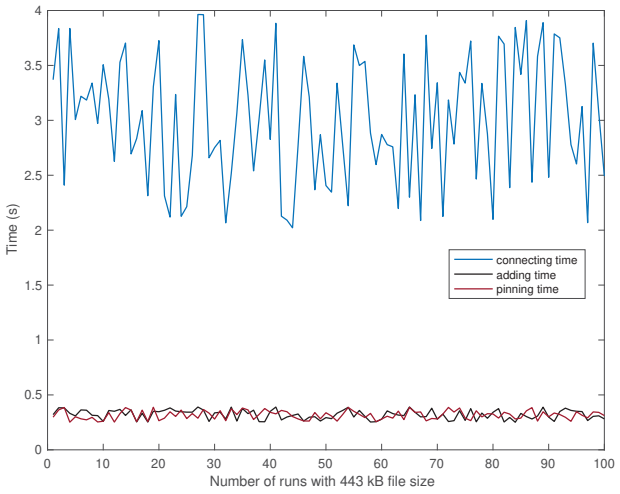

Fig. 6. T_{IPFS} including functions; adding (V=0.0018 s), pinning (V=0.0016 s), and connecting to IPFS daemon (V=0.3305 s)

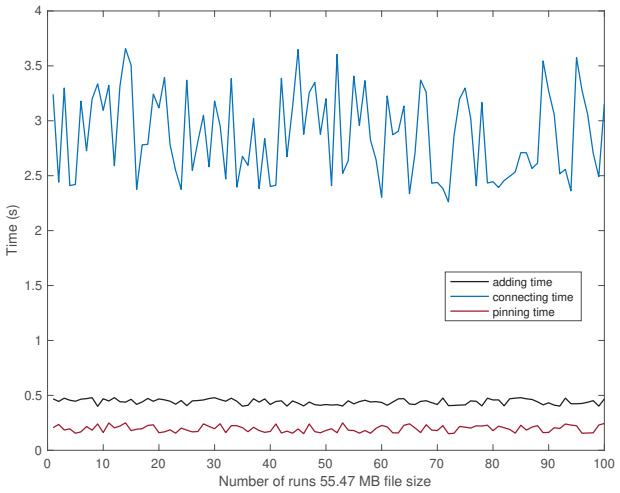

Fig. 7. T_{IPFS} including functions; adding (V=0.005 s), pinning (V=0.0089 s), and connecting to IPFS daemon (V=0.150 s)

VI. Conclusion and Future Work

Data storage is an essential requirement for applications that help satisfy the digital world's expected requirements.

TABLE I
PROPERTIES FOR DIFFERENT SCENARIOS

Cases	Storage	Time	Performance
Blockchain	Low	Low	Low
Scenario 1	High	High	Medium
Scenario 2	High	Medium	High

Performance issues and data storage capacity should therefore be given attention. In this paper, we presented a combined solution based on a public blockchain and a DHT. We started the paper by explaining the blockchain and smart contract solutions. Furthermore, we discussed how the blockchain manages transactions by its immutable historical data. Storing and sharing data in a DHT or the IPFS were discussed. This system can be beneficial for storing decentralized data in parallel systems.

Finally, this paper illustrated two scenarios of data storage immutability concerning storage of bulky mutual agreements in IPFS and blockchain systems and showed the blockchain encounter. Although blockchain is a safe technology to keep transaction data, the risk of security attacks remains. For future work we consider adding methods for strong identities of parties entering agreements and securing the system against common threats. We also consider extending our study to include multi-party agreements.

Acknowledgment

This work has been supported by the Kolarctic CBC under grant KO4096.

References

[1] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *International Journal of Web and Grid Services*, vol. 14, no. 4, pp. 352–375, 2018.

[2] J. Liu, B. Li, L. Chen, M. Hou, F. Xiang, and P. Wang, "A data storage method based on blockchain for decentralization dns," in *IEEE Third International Conference on Data Science in Cyberspace (DSC)*, pp. 189–196, IEEE, 2018.

[3] M. Alizadeh, K. Andersson, and O. Schelén, "A Survey of Secure Internet of Things in Relation to Blockchain," *Journal of Internet Services and Information Security*, vol. 3, no. 10, pp. 47–75, 2020.

[4] M. H. ur Rehman, K. Salah, E. Damiani, and D. Svetinovic, "Trust in blockchain cryptocurrency ecosystem," *IEEE Transactions on Engineering Management*, 2019.

[5] V. L. Lemieux, "Blockchain and distributed ledgers as trusted record-keeping systems: An archival theoretic evaluation framework," in *Future Technologies Conference (FTC)*, vol. 2017, pp. 1–11, 2017.

[6] M. Conti, E. S. Kumar, C. Lal, and S. Ruj, "A survey on security and privacy issues of bitcoin," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3416–3452, 2018.

[7] F. Tschorsch and B. Scheuermann, "Bitcoin and beyond: A technical survey on decentralized digital currencies," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 2084–2123, 2016.

[8] I.-C. Lin and T.-C. Liao, "A survey of blockchain security issues and challenges.," *International Journal of Network Security*, vol. 19, no. 5, pp. 653–659, 2017.

[9] A. Reyna, C. Martín, J. Chen, E. Soler, and M. Díaz, "On blockchain and its integration with IoT. challenges and opportunities," *Future Generation Computer Systems*, vol. 88, pp. 173–190, 2018.

[10] E. Politou, E. Alepis, C. Patsakis, F. Casino, and M. Alazab, "Delegated content erasure in IPFS," *Future Generation Computer Systems*, vol. 112, pp. 956–964, 2020.

[11] M. S. Ali, K. Dolui, and F. Antonelli, "IoT data privacy via blockchains and IPFS," in *Proceedings of the Seventh International Conference on the Internet of Things*, pp. 1–7, 2017.
[12] H. R. Hasan and K. Salah, "Proof of delivery of digital assets using blockchain and smart contracts," *IEEE Access*, vol. 6, pp. 65439–65448, 2018.
[13] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A survey on network codes for distributed storage," *Proceedings of the IEEE*, vol. 99, no. 3, pp. 476–489, 2011.
[14] Z. Li, H. Wu, B. King, Z. B. Miled, J. Wassick, and J. Tazelaar, "On the integration of event-based and transaction-based architectures for supply chains," in *2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, pp. 376–382, IEEE, 2017.
[15] A. ul Haque, M. S. Ghani, and T. Mahmood, "Decentralized transfer learning using blockchain & IPFS for deep learning," in *2020 International Conference on Information Networking (ICOIN)*, pp. 170–177, IEEE, 2020.
[16] M. Shah, M. Shaikh, V. Mishra, and G. Tuscano, "Decentralized cloud storage using blockchain," in *2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184)*, pp. 384–389, IEEE, 2020.
[17] M. Steichen, B. Fiz, R. Norvill, W. Shbair, and 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)*, pp. 1499–1506, IEEE, 2018.
[18] J. Sun, X. Yao, S. Wang, and Y. Wu, "Non-repudiation storage and access control scheme of insurance data based on blockchain in IPFS," *IEEE Access*, vol. 8, pp. 155145–155155, 2020.
[19] A. Tenorio Fornés, S. Hassan, and J. Pavón Mestras, "Peer-to-peer systems design trade-offs: a framework exploring the balance between blockchain and IPFS," *Annals of Telecommunications*, 2020.
[20] I. I. Yiakoumis, M. E. Papadonikolakis, H. E. Michail, A. P. Kakarountas, and C. E. Goutis, "Maximizing the hash function of authentication codes," *IEEE Potentials*, vol. 25, no. 2, pp. 9–12, 2006.
[21] P. Maymounkov and D. Mazieres, "Kademlia: A peer-to-peer information system based on the XOR metric," in *International Workshop on Peer-to-Peer Systems*, pp. 53–65, Springer, 2002.
[22] M. Masse, *REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces*. O'Reilly Media, Inc., 2011.