Accredited Ranking SINTA 2
Decree of the Director General of Higher Education, Research and Technology, No. 158/E/KPT/2021
Validity period from Volume 5 Number 2 of 2021 to Volume 10 Number 1 of 2026

Published online on: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 7 No. 3 (2023) 670 - 680 ISSN Media Electronic: 2580-0760

Secure Cybersecurity Information Sharing for Sectoral Organizations Using Ethereum Blockchain and IPFS

Tony Haryanto[1], Kalamullah Ramli[2]
[1,2]Department of Electrical Engineering, Faculty of Engineering, Universitas Indonesia
[1]tony.haryanto@ui.ac.id, [2]kalamullah.ramli@ui.ac.id

Abstract

The COVID-19 pandemic has resulted in increased cross-sector cyber-attacks. Passive and reactive cybersecurity techniques relying solely on technology are insufficient to combat sophisticated attacks, necessitating proactive and collaborative security measures to minimize attacks. Cybersecurity Information Sharing (CIS) enhances security via proactive and collaborative cybersecurity information exchange, but its implementation via cloud services faces threats from man in the middle (MITM) and distributed denial of service (DDoS) attacks, as well as a vulnerability in cloud storage involving centralized data control. These threats and vulnerabilities result in a lack of user confidence in the confidentiality, integrity, and availability of information. This paper proposes Secure Cybersecurity Information Sharing (SCIS) to secure Cybersecurity Information in sectoral organizations using the private interplanetary file system (IPFS) network and the private Ethereum Blockchain network. Private Ethereum Blockchain enables secure and transparent transaction logging, while Private IPFS network provides decentralized storage, addressing vulnerabilities in centralized storage systems. The outcomes of the tests reveal that the suggested SCIS system offers cybersecurity information availability, confidentiality, and integrity. SCIS provides a high level of security to protect cybersecurity information exchanged between sectoral organizations using the Private Ethereum Blockchain network and the Private IPFS network so that organizations can safely share and utilize information.

Keywords: cybersecurity; information; sharing; blockchain; IPFS.

1. Introduction

Since COVID-19 spread around the world, The number of cyber-attacks has increased across many sectors. Such threats include increasing types and methods of carrying out attacks [1]. The increasing varieties of cyber-attacks explain that applying cybersecurity techniques that rely only on technology is insufficient to protect against sophisticated cyber-attacks[2]. Understanding security techniques is shifting from passive to reactive security, where the organization initially passively waits for an attack and becomes active after an attack [3]. However, reactive security is also not enough to minimize the occurrence of attacks because reactive security schemes cannot prevent attacks [4]. Therefore, proactive security is needed, a security scheme that will proactively continuously look for cybersecurity information that may occur long before the attack occurs. One way it can be used is to share cybersecurity information from within or outside the organization [5].

Cybersecurity Information Sharing (CIS) is a collaborative effort between organizations to address cybersecurity challenges quickly and precisely for all sectors, nationally and internationally [6]. Collaboration in adopting CIS between organizations is a process that benefits all involved organizations [7]. For example, other organizations with similar characteristics can learn from victims to avoid similar attacks [8]. Cybersecurity Information Sharing in organizations can assist stakeholders in utilizing cybersecurity resources to make informed decisions in formulating defense techniques, cyber threat detection, strategy, and mitigation [9]. Stakeholders who receive cyber security information then use it to improve their security and can protect other stakeholders by sharing information to prevent these cyber threats from spreading [10]. In practice, however, CIS is challenging because various threats can influence the infrastructural components of an organization, such as the internet network, communication, and storage [11]. Traditional storage management system involves a central authority to control large amounts of data. One cannot trust data's confidentiality, integrity, and authenticity [12]. A single point of failure for storage systems that offer cybersecurity data is caused by attacks using distributed

Accepted: 22-02-2023 | Received in revised: 14-05-2023 | Published: 02-06-2023

denial of service (DDoS), which is one of the many issues with centralized storage systems [13]. This attack causes the CIS service on centralized storage to be unavailable. The next attack is the man in the middle (MITM) [14] which attacks the transmission of cybersecurity information which causes unauthorized entities to receive cybersecurity information and causes data leakage. These entities can damage the integrity of the data and modify the data, which is then passed back to the recipient so that the recipient receives the modified data. Hence the need for distributed and decentralized technology that guarantees information security.

The blockchain is a decentralized peer to peer network that consists of numerous nodes, which are individual computers [15]. Blockchain development has become one of the most popular technological innovations recently first revealed in 2008 and is commonly known as Bitcoin [16]. Blockchain has several unique characteristics, including decentralized, distributed, secure, transparent, and immutable [17]. One type of blockchain often used in developing decentralized applications is Ethereum. Ethereum is a distributed ledger platform that enables developers to create decentralized applications (also known as DApps) and smart contracts [18]. In 2013, Vitalik Buterin built Ethereum, released to the public in 2015. Like Bitcoin, Ethereum is a decentralized network that allows users to send and receive value without intermediaries [19]. However, unlike Bitcoin, Ethereum is not just a digital currency but a place where developers can make autonomous apps and smart contracts.

A self-executing contract is referred to as a smart contract [20]. In this type of contract, the terms of the agreement between the sender and recipient are encoded directly into the line of code. Smart contracts enable the automated execution of transactions [21]. They can facilitate various applications, from digital identities and asset tracking to voting and decentralized finance (DeFi) systems [17]. Ethereum's native cryptocurrency is Ether (ETH), used to pay gas fees on the network. The Ethereum blockchain can be divided into public and private networks[22]. The Public Ethereum is a decentralized, open source blockchain network that can be accessed by anyone with an internet connection [23].

This network sends and receives Ether and executes smart contracts and decentralized applications (DApps). Anyone can participate in the network by running nodes, mining, or contributing to the development of the ecosystem [18]. A private Ethereum network is a blockchain network operated and controlled by a single organization or individual. It is used to create closed ecosystems for specific purposes, such as testing smart contracts or running private supply chain networks. Private Ethereum networks have more control over network rules, privacy, and security and are not accessible to the public [24]. Private Ethereum networks can utilize several consensus processes to confirm transactions and protect the network, such as Proof of Authority (PoA) or Proof of Stake (PoS) [25]. Overall, the private Ethereum blockchain offers organizations greater control and customization over their blockchain network, allowing them to create more secure and customized systems for their specific needs [26]. A distributed peer to peer system called Interplanetary File System (IPFS) can store much hypermedia. IPFS (InterPlanetary File System) [27] is a distributed file system and hypermedia protocol designed to one day replace the World Wide Web. The Merkle Directed Acyclic Graph (DAG) is managed using a block storage paradigm with hypertext linkages [28]. Since IPFS is distributed, there is not a single failure point. There are many problems with HTTP, such as being slow, not having versions from the past, and being centralized. Thus, IPFS overcomes the weakness of HTTP.

This paper presents a solution for securely sharing cybersecurity information built on Private IPFS and Private Blockchains. Files uploaded on IPFS proxies are stored on a Private IPFS Network using the decentralized file distribution system of IPFS. In addition, we use Ethereum smart contracts and blockchain as traceable servers to record file uploads and share information on the network to avoid information corruption. The combination of $ and Private Blockchain networks creates a secure cybersecurity information sharing (SCIS) system that provides a high level of security to protect cybersecurity information so that sectoral organizations can safely share and utilize cybersecurity information.

The remaining sections are organized as follows. This study's methodology is discussed in Section 2. Section 3 contains the findings of applying this method. Finally, Section 4 discusses the conclusions that can be derived from this study's findings.

2. Research Methods

In this study, the Software Development Life Cycle (SDLC) research method was chosen in the design, implementation, and testing of the SCIS system [29]. SDLC is a methodology used in software development. The analysis phase is the initial stage of the SDLC, where system and user requirements are collected and analyzed to identify the features and functions needed in the software. After the analysis phase, the design phase begins, where the system architecture, system scenario, and system test scenario are planned. The next stage is implementation, where the software is built and coded based on the agreed design. The testing phase is an important stage in the SDLC, where the software is tested according to the test scenarios that have been

DOI: https://doi.org/10.29207/resti.v7i3.4956

created to ensure that it functions correctly and meets user requirements. After the testing phase is complete, the software is evaluated to assess functionality and ensure that the system meets security requirements. In the SDLC as shown in Figure 1, these stages are interrelated and interdependent on one another to ensure quality and efficient software development [29].

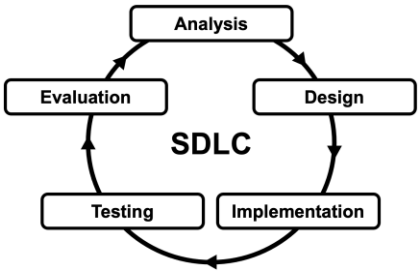

Figure 1. Software Development Life Cycle (SDLC)

3. Results and Discussions

The obtained results will be presented following the SDLC model, which comprises various stages such as analysis, design, implementation, testing, and evaluation. each stage will be described in detail along with related tables and figures.

3.1. System Analysis

System analysis in this study is about understanding the problem being handled and how the proposed solution works. The problem is availability, integrity, and confidentiality in implementing cybersecurity information sharing among sectoral organizations. The proposed solution is to make use of the Ethereum Blockchain and the Interplanetary File System to produce a secure and decentralized platform for sharing cybersecurity information. The system will work by enabling users to securely upload and share information on the platform. The use of Blockchain ensures that the data is immutable and cannot be altered or deleted without authorization. IPFS, on the other hand, ensures that information is decentralized and distributed across multiple nodes, making it more resistant to attacks. The system will be accessible to authorized users, and access to information will be restricted based on user roles and permissions. The system will also incorporate encryption to ensure that data is protected in transit and at rest. Overall, the proposed system offers a promising solution to the security problem of cybersecurity information sharing in sectoral organizations. By leveraging the Ethereum Blockchain and IPFS, the system provides a decentralized and tamper-resistant platform that ensures the privacy and security of shared information. Table 1 is the system category.

Table 1. System Category

Category	Description
Problem	Availability, integrity, and confidentiality in implementing cybersecurity information sharing among sectoral organizations.
Solution	A secure and decentralized platform for sharing cybersecurity information using Ethereum Blockchain and Interplanetary File System (IPFS).
Features	Users can securely upload and share information, information stored on Blockchain is incorruptible, IPFS makes information decentralized, access to information restricted based on user roles and permissions, and encryption to protect data in transit and at rest.
Security	Information is secure from tampering and distributed across multiple nodes for improved resilience, access to information restricted to authorized users, and encryption protects data.
Accessibility	Only authorized users can access the platform and information, the platform is accessible remotely and user-friendly.
Advantages	Decentralized and secure, improves collaboration and coordination among sectoral organizations, ensures the privacy and security of shared information, and helps address the problem of cybersecurity information sharing.
Limitations	Requires good internet access and incurs costs for running a Blockchain network.

3.2. System Design

The suggested system architecture for SCIS is shown in Figure 2. Each user must be connected to a decentralized application called SCIS DApp, a private IPFS network, and a private Ethereum blockchain to interact with this system. SCIS DApp users consist of data owners and data users, while administrators are responsible for user registration. Each user has a peerID for accessing the IPFS network and an Ethereum address for exchanging data on the Ethereum blockchain.

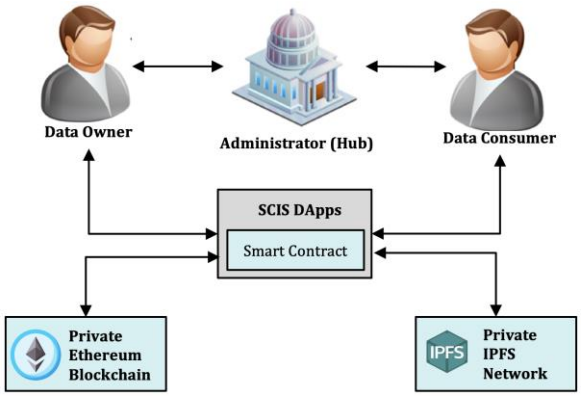

Figure 2. The System Architecture of The Proposed SCIS

IPFS and Ethereum blockchain are regulated as private networks to maintain cybersecurity information confidentiality. The cybersecurity information files are only distributed between data owners and data users. Additionally, only they will be able to store and retrieve file hashes. For the SCIS DApp to work on regular browsers like Google Chrome or Mozilla Firefox, users

DOI: https://doi.org/10.29207/resti.v7i3.4956

must install an Ethereum wallet like Metamask [30]. Additionally, an Ethereum wallet must import the user's private Ethereum blockchain Ethereum address. The SCIS DApp includes a smart contract with commands for storing hash values on a private Ethereum blockchain. The SCIS requires a system that includes entities and their roles. Accordingly, the SCIS system defines five entities, according to Table 2.

Table 2. Entities and function SCIS system

Entity	Function
Data Owner (DO)	Entities that can store information in IPFS and record it on the blockchain. Data owners can also share data stored in IPFS with Data Users using the SCIS DApp
Data Consumer (DC)	Entities that can consume the Data Owner's information that has been shared by the Data Owner using the SCIS DApp
Administrator	The entity responsible for enrolling all users into the IPFS private network and the private Ethereum blockchain
SCIS DApp	Cybersecurity information store and sharing simulation platform connected to smart contract technology, private IPFS network, and private Ethereum blockchain
Private IPFS Network	The private network connecting each entity uses the same swarm key within the same IPFS network
Private Ethereum Blockchain	A private network that connects each entity using a different private key in the same network and provides blocks of related information through hash values as transaction records in the private Ethereum blockchain

3.3. Smart Contract Deployment

Smart contracts are software applications that enable the automatic execution of transactions on the blockchain network, facilitating agreements between two or more parties without the need for an intermediary. These contracts, once deployed on the blockchain, become immutable and tamper-proof, providing a high degree of security and reliability. This makes smart contracts an ideal tool for executing agreements between parties without the need for trust, as the terms and conditions of the agreement are encoded into the contract and enforced by the blockchain network. [17].

Figure 3 describes the smart contract deployment process which involves several steps [31].

Create smart contract code in a high-level language for building smart contracts on the Ethereum blockchain, like Solidity.

Compile code for smart contracts using a compiler like the Solidity compiler. This technique yields bytecode and Application Binary Interface (ABI). After that Create an Ethereum account and get Ether, the Ethereum blockchain's native cryptocurrency. Ether must cover the transaction fees connected with the deployment of smart contracts.

Deploy smart contracts by creating and sending transactions with the required bytecode and transaction data. This may be accomplished via an Ethereum wallet or the command line interface and then verify a smart contract's blockchain implementation using transaction hash, contract address, bytecode, and contract's ABI, or application binary interface.

Once a smart contract is used, it can be dealt with using various tools and interfaces, such as the JavaScript library web3.js, which connects with Ethereum-based apps [31].

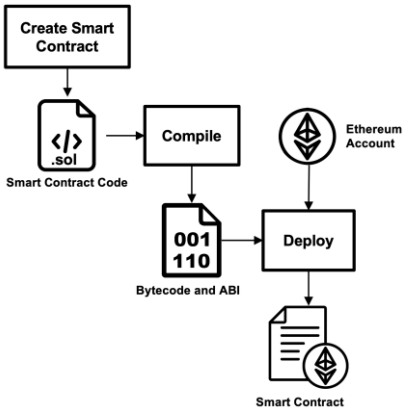

Figure 3. Smart Contract Deployment Process

3.4. System Scenario

The registration process for the private IPFS network is illustrated in Figure 4. In this scenario, the administrator serves as the boot node, and all SCIS DApp application users must register by providing their organization's original identity to the administrator. Once verified, the administrator will generate a unique swarm key and distribute it to all registered users. This swarm key ensures that all users are authenticated on the same private IPFS network. Additionally, users must configure the bootstrap node configuration using the peerID and IP address of each boot node to ensure proper communication and connectivity within the network.

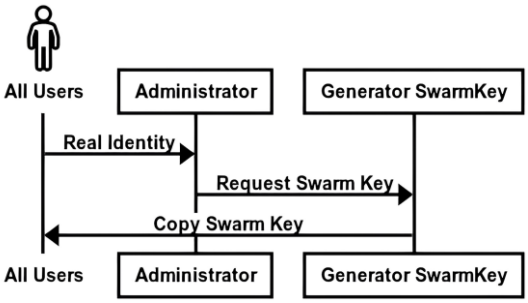

Figure 4. Private IPFS Network Registration Sequence

3.4.2 Private Ethereum Blockchain Registration

The process for obtaining an account to perform transactions on the SCIS DApp application is illustrated

DOI: https://doi.org/10.29207/resti.v7i3.4956

in Figure 5. To register, all users must submit their organization's real identity to the administrator. Upon verification, the administrator will generate a unique key pair for each user by entering a passphrase. The key pair consists of a public key that serves as a shared account address and a private key that acts as the user's identity for creating an Ethereum Wallet account. It is important to note that only the private key owner can access their account, ensuring a secure and private user experience.

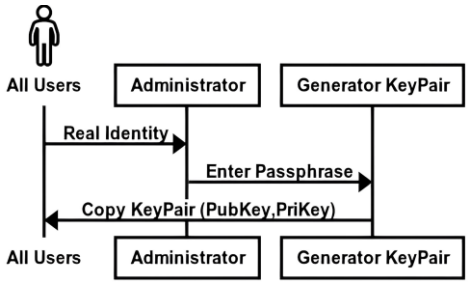

Figure 5. Private Blockchain Registration Sequence

As shown in Figure 6, the file uploading process on the SCIS system involves multiple steps. First, the Data Owner (DO) uploads the files to the SCIS DApp, which then transmits them to the secure internal network of IPFS. The files are stored in a distributed manner on the private IPFS network, and IPFS sends the hash value of the uploaded file as proof of successful storage. Additionally, the SCIS DApp records the file's hash value on the private Ethereum blockchain, ensuring that the file's integrity is maintained and can be verified at any point in the future.

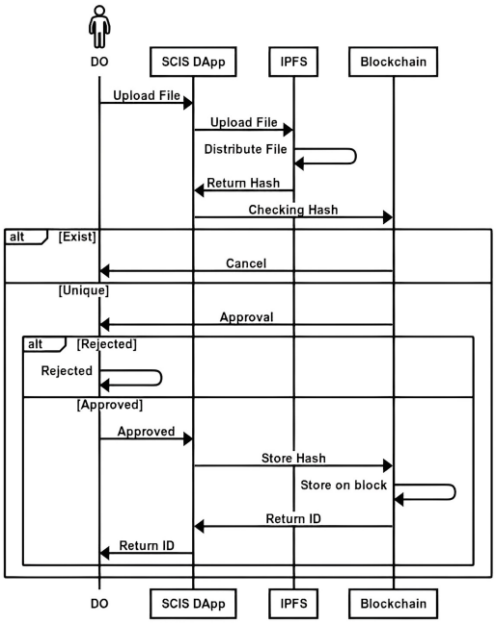

Figure 6. Sequence Diagram of Storing a File

In the private Ethereum blockchain, a check will be carried out to determine whether the hash value has been stored. If the hash value exists, the transaction cannot be continued because the hash value has already been recorded in the private Ethereum blockchain. If the hash value is Unique, the transaction can be continued by sending a storage agreement to the data owner. The approval is a Gas Fee to be charged to the Data Owner. If the Data Owner refuses, then recording on the blockchain network cannot be done. If the Data Owner agrees, store the hash in the private Ethereum blockchain. In the Ethereum private blockchain, the hash value will be stored in a transaction block that all members know of the private Ethereum blockchain. After the hash is recorded in the blockchain, the private Ethereum blockchain will send the Transaction Identity to the Data Owner as proof that the hash value has been stored in the private Ethereum blockchain.

Figure 7 depicts the file-sharing capabilities of a SCIS system. The data owner (DO) uploads the file's hash value to be sent and the data consumer (DC) address as the data receiving entity into SCIS DApp. SCIS DApp will transmit a hash value to the private network of IPFS. After then, the hash value is checked by the private IPFS network. Check a hash to determine whether the file is stored in a private IPFS. If the hash is unique, then the file to be sent by data owner is not stored in private IPFS network, or file to be sent is not in the IPFS network. If the hash exists, the next step is that IPFS will download the file and send it to SCIS DApp, which will be distributed to the data consumer according to the address of the consumer data uploaded by the data owner in the previous stage.

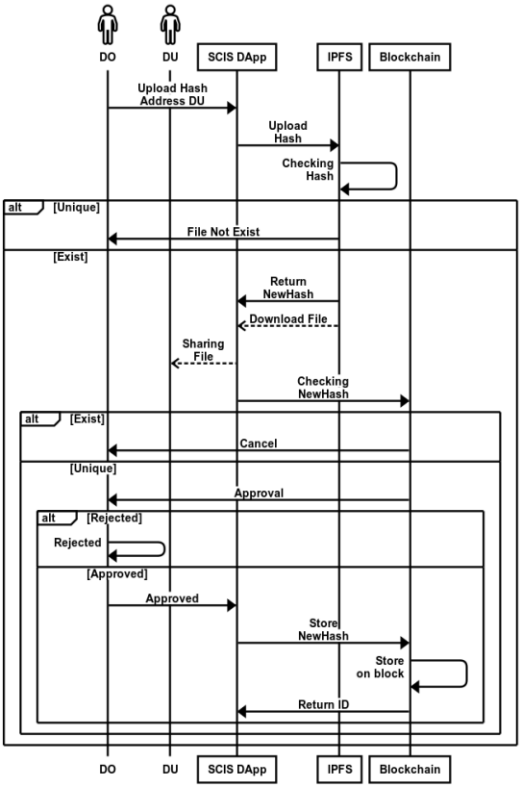

Figure 7. The SCIS System File Sharing Sequence Diagram

DOI: https://doi.org/10.29207/resti.v7i3.4956

After that, IPFS will also return a new hash value (NewHash) to prove the download process has succeeded. This NewHash value will be saved in the Ethereum private blockchain. In the private Ethereum blockchain, a check will be carried out to determine whether the NewHash value has been stored in the private Ethereum blockchain. If NewHash exists, the transaction cannot be continued because the hash value has already been recorded in the private Ethereum blocks. If NewHash is Unique, the transaction can be continued by sending a storage agreement to the data owner. The approval is a Gas Fee, which will be charged to data owner. If data owner refuses, recording into private Ethereum blockchain cannot be done. If the data owner agrees, save the NewHash into the private Ethereum blockchain. Within the private Ethereum blockchain, the Newhash value will be stored in a transaction block known to all private Ethereum members. After the NewHash is recorded in the blockchain, the private Ethereum blockchain will send the Transaction Identity to the data owner as proof that the NewHash value has been stored in private Ethereum blockchain. The Transaction Identity is a crucial log for the data owner, as it guarantees the veracity and safety of the information provided to the end user.

3.5. Testing Scenarios

Scenario of File Availability Testing in Figure 8 shows an availability test scenario using four nodes on a private IPFS network that shares the same swarm key and operates in a VMware simulation environment. Each IPFS node will have a different node name and internet protocol address but still be connected to the same network. Each node's PeerID, which is used to interact on the private IPFS network, serves as a means of identification for all nodes. To test file availability, we will experiment with downloading information when one of the nodes is down.

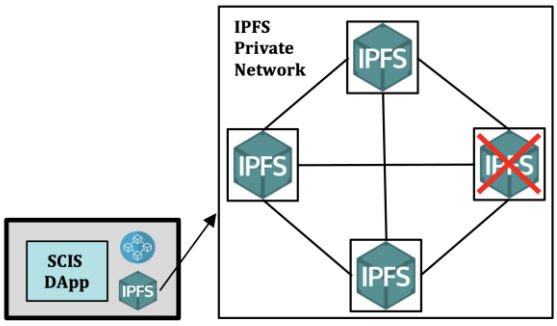

Figure 8. The Scenario of File Availability Testing

To conduct the scenario for testing file integrity depicted in Figure 9, the authors began by uploading File A from user A and File B from user B into the CIS DApp. Despite being similar in size, there was a slight difference in their contents. The experiment was designed to analyze the hash value output from File A and File B in the CIS DApp, to determine if the changes in their contents had an impact on the hash value. By conducting this experiment, the authors aimed to ensure the integrity of the data being stored and shared through the CIS DApp and to ensure that any changes to the data could be immediately detected and addressed. The results of this experiment are presented and discussed in detail in the subsequent section of the paper.

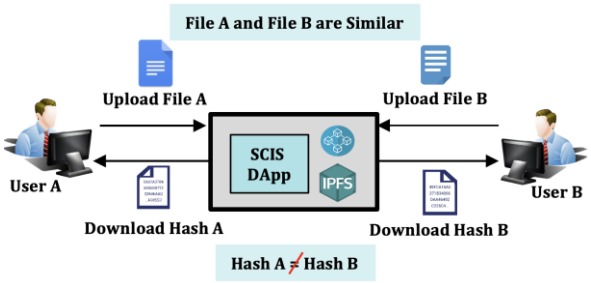

Figure 9. The Scenario of File Integrity Testing

As shown in Figure 10, the confidentiality test involves downloading files from client nodes that are located outside private IPFS network or public IPFS network and are not registered users. This test is conducted to assess the security and confidentiality of the private IPFS network. To perform this test, the public IPFS network is used, which has a different swarm key than the swarm key of the private IPFS network. However, the hash value of the file that is downloaded from the private IPFS network is valid and can be used to verify the integrity of the downloaded file. This test ensures that the private IPFS network is secure and that files stored on it cannot be accessed or downloaded by unauthorized users, providing a high level of confidentiality and security to the data owners.

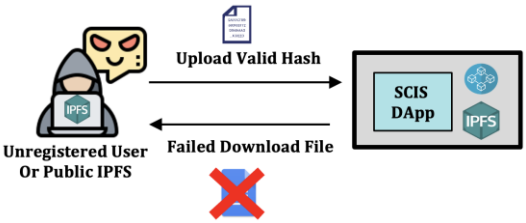

Figure 10. The Scenario of File Confidentiality Testing

3.6. Implementation

SCIS DApp is a front-end system facilitating interaction between users with private IPFS networks and the Private Ethereum Blockchain. The SCIS DApp is built in the Truffle development environment and uses JavaScript. SCIS DApp contains a smart contract with the filename SCISSmartContract.sol, created in the solidity programming language. SCIS DApp has the main function of storing file hashes in the Ethereum blockchain private network. In deploying a smart contract, the administrator must have an Ethereum account by generating a key pair on the Ethereum blockchain private network. The private key will create an Ethereum wallet (Metamask) account with a predetermined number of Ether testers. After getting an

DOI: https://doi.org/10.29207/resti.v7i3.4956

account, deploy the smart contract using truffle migrations. The SCIS smart contract deployment results describe the successfully deployed smart contract, such as transaction hash, account address, contract address, gas used, block number, and block timestamp as shown in figure 11. The contract address is the identity of the smart contract, which is used as the address for interacting between the user and the smart contract

```
Deploying 'SCISSmartContract'
-----------------------------
> transaction hash:    0x25b8a881cf9f3f3d62c963a2762468fd
> Blocks: 0            Seconds: 0
> contract address:    0x7437D3171ac9Bd7f352485F1412D7a0A
> block number:        3
> block timestamp:     1630692596
> account:             0x90f8bf6a479f320ead074411a4b0e794
> balance:             99.98395896
> gas used:            190936
> gas price:           20 gwei
> value sent:          0 ETH
> total cost:          0.00381872 ETH

> Saving migration to chain.
> Saving artifacts
-------------------------------------
> Total cost:           0.00381872 ETH
```

Figure 11. SCIS Smart Contract

Private IPFS Network Registration in Figure 12 simulates registration on a private IPFS network. The administrator acts as a boot node with node name ipfs1 and IP address 172.16.55.148, data owner (DO) with node name ipfs2 and IP address 172.16.55.149 then consumer data (DC) with node name IPFS and IP address 172.16.55.131. All users using the SCIS DApp must request a swarm key by submitting the original identity of the organization to the administrator. Then the administrator will generate the swarm key with the swarm key generator and give the same swarm key to all registered users. The swarm key authenticates that all users are on the same private IPFS network. The bootstrap node must then be configured by each user using each boot node's peerID and IP address.

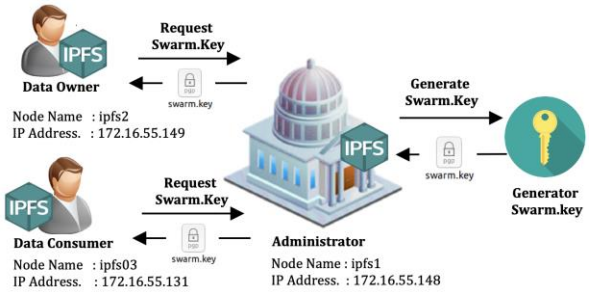

Figure 12. Private IPFS Network Registration

Table 3. Result of Implementing Registration on The Private IPFS Network

Role	Node	IP Address	Swarm Key
Admin	Ipfs1	172.16.55.148	4be3a22ff97ebd0e4d e60a8c321b02de
DO	Ipfs2	172.16.55.149	4be3a22ff97ebd0e4d e60a8c321b02de
DC	Ipfs03	172.16.55.131	4be3a22ff97ebd0e4d e60a8c321b02de

Table 3 is the result of implementing registration on the private IPFS network. Each node gets the same swarm key to communicate with each other in the same private IPFS network.

Figure 13 simulates the registration process on the private Ethereum blockchain. All users making transactions on the SCIS DApp simulation platform must request a key pair by submitting their real organizational identity to the administrator. Then the administrator will generate a key pair by entering passphrases into the ganache. Ganache is an Ethereum network simulator built to facilitate the creation of smart contracts for use on the Ethereum network. [32]. Each user will receive a different key pair. The public key acts as a public account address known to the public, and the private key acts as an identity in creating an Ethereum Wallet (Metamask) account that only the private key owner knows.

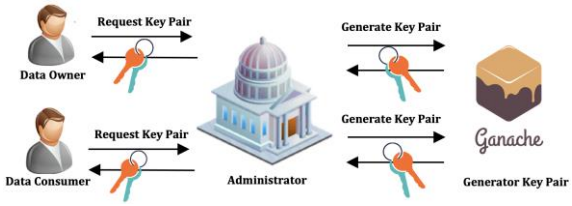

Figure 13. Registration on Private Ethereum Blockchain

Figure 14 shows the owner data key pair, and Figure 15 shows the consumer data key pair. Both key pairs result from key pair generation by the ganache key pair generator. The key pair consists of an account address as a public and private key. Private keys generated in ganache cannot be used in public blockchains. The private key is only to be used for development and simulation. The key pair generation process is an essential part of secure communication in the decentralized system. The use of the ganache key pair generator is a common practice in developing blockchain applications for testing purposes.

Figure 14. Key Pair Data Owner

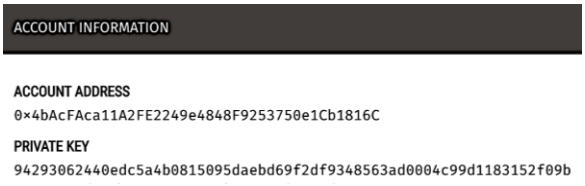

Figure 15. Key Pair Data Consumer

Storing a File on SCIS System is demonstrated in Table 3, the system executes the Submit function in the

DOI: https://doi.org/10.29207/resti.v7i3.4956

SCISSmartContract.sol file to store files to the private IPFS network and send log transactions to the private Ethereum blockchain network. Prior to storage on the private IPFS network, files are converted into a buffer (array of data). After being successfully stored on IPFS, IPFS returns the hash value. This causes Metamask to verify the transaction. Upon confirmation, the private Ethereum blockchain network begins mining to include the transaction in the next available block. If the process is successful, private Ethereum blockchain will return the Transaction ID as proof that the transaction recording process was successful. If it fails, then the process is finished.

Table 4. The File-Storing Algorithm

Algorithm 1
1: function Submit
2: get smart contract address
3: convert file content to buffer
4: call method *store()*
5: send file buffer to ipfs
6: return hash value
7: put hash value into blockchain
8: call method *log()*
9: send hash into *log()*
10: if *log()* is success
11: return transactionid
12: else
13: return error
14: end if
15: end function

Figure 16 shows the transaction log resulting from the storing file in the SCIS System. The process of storing files on the SCIS system begins with the data owner uploading files to the SCIS DApp, the data will be saved on the private IPFS network, and the storage transaction will be logged on the private Ethereum in a block. A transaction log simulation was conducted on ganache using network id 5777 and Remote Procedure Call Server 127.0.0.1:7545. The transaction result contains the transaction block number, gas used, gas limit, mining time, block hash, transaction hash, data owner address, and smart contract address.

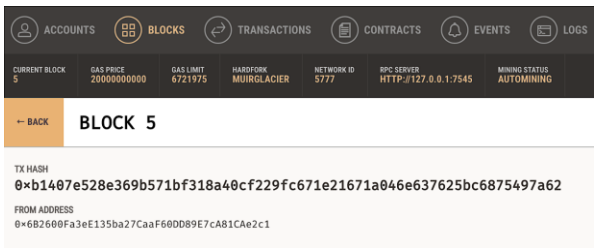

Figure 16. Storing File Log Transactions on Blockchain

Sharing a file on SCIS System is shown in Table 5, the system calls the Submit function in SCISSmartContract.sol to share files with consumers and send transaction logs to the private Ethereum blockchain network. To obtain a file from the private IPFS network and then distribute it to a user of the network, The owner of the file first sends the file's hash number into the network. The hash number is a unique identifier for the file that lets it be found on the network and retrieved. IPFS returns a newhash value in addition to the file from the uploaded file's hash to the private IPFS network. To confirm the transaction, Metamask is triggered by the newhash. As soon as a transaction is successfully confirmed, the private Ethereum blockchain network starts the mining procedure for storing the transaction's details in the next block. The Ethereum blockchain private network will produce a Transaction ID as evidence that the transaction recording process was accomplished if it is successful. If it fails, then the process is finished.

Table 5. The File-Sharing Algorithm

Algorithm 2
1: function Submit
2: get consumer address
3: get smart contract address
4: call method *retrive()*
5: send hash of file into ipfs
6: return file, newhash value
7: send file to consumer address
7: put newhash value into blockchain
8: call method *log()*
9: send hash into *log()*
10: if *log()* is success
11: return transactionid
12: else
13: return error
14: end if
15: end function

Figure 17 shows the transaction log resulting of the sharing file process on the SCIS System by the data owner to the consumer data. The file-sharing process on the SCIS system begins with the data owner uploading the valid hash value of the file and consumer data address into the SCIS DApp. The hash value will be transmitted to the IPFS private network. Suppose the file is associated with a hash value within the private IPFS network. The file will be transferred to the consumer's data address, and the file-sharing transaction will be recorded on the private Ethereum blockchain. A transaction log simulation was conducted on ganache using network id 5777 and Remote Procedure Call Server 127.0.0.1:7545. The transaction result contains the transaction block number, gas used, gas limit, mining time, block hash, transaction hash, data owner address, smart contract address, and data consumer address.

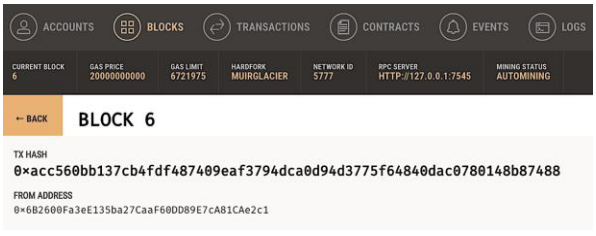

Figure 17. Sharing File Log Transactions on Blockchain

DOI: https://doi.org/10.29207/resti.v7i3.4956

3.7. Testing

File Availability Testing was conducted in a VMware simulation environment using four nodes on a private IPFS network that share the same swarm key. Each IPFS node has a different IP address but is still on the same network. PeerID is a unique identifier for each node in a private IPFS network that all node shares. Table 6 shows the attempted upload and download of information from a private IPFS network. The ipfs2 node acts as a node that uploads data to the private IPFS network, while the other nodes act as nodes that download information from the private IPFS network. Table 6's experimental results demonstrate that the suggested SCIS system's upload and download functionalities can work as intended.

Table 6. Upload and Download Functionality System

Node	IP Address	Status	Testing	Result
Ipfs1	172.16.55.149	Active	Download	Available
Ipfs2	172.16.55.148	Active	Upload	Available
Ipfs3	172.16.55.139	Active	Download	Available
Ipfs4	172.16.55.138	Active	Download	Available

The experiment's findings, which were used to confirm the availability of files, are shown in Table 7. In this experiment, we attempted to download data while one of the nodes was down, and the results indicate that other nodes can provide the required information. This experiment proves the fault-tolerance capability of the system, ensuring the availability of data even if one of the nodes is not operational. The results indicate that the system can recover from a node failure, and the other nodes can take over its functions, ensuring that data remains available. This is a crucial aspect of any distributed system, and the experiment confirms that the SCIS system is designed with high availability in mind.

Table 7. Results of Availability Testing

Node	IP Address	Status	Testing	Result
Ipfs1	172.16.55.149	Active	Download	Available
Ipfs2	172.16.55.148	Active	Upload	Available
Ipfs3	172.16.55.139	Active	Download	Available
Ipfs4	172.16.55.138	Down	Download	Not available

The hash function is a function that can change data into other data that has a specific size according to the algorithm it uses. Due to its unique nature, the hash function can check the similarity between two files. If the two files are the same, the resulting hash value should be the same, and if the two files are different, the resulting hash value should be different. This test is performed by uploading two files with similar names and sizes (5,510 bytes or 5.5 kilobytes different) but with minor modifications in their content. SCIS DApp identified both files with different hashes. The SCIS DApp feature can verify the integrity of each file. This feature can help the user that if there is a modification to the file made by another user, it will result in a change in the integrity of the file, as evidenced by a different hash value. This point is shown in table 8.

Table 8. Results of Integrity Testing

Name File	Volume	Transaction ID	Hash File
CI_Jengking Merah.pdf	1.711.649 bytes	0x85806dd0e65 ebb700b18a25d 065a18b953c6d f703eb23df4456 7ec062006b0c6	Qmccehyfh aQmpP35D 91a7G25i3 yKfdzro6R H9mhGnW vA3n
CI_Jengking Merah [Mod].pdf	1.717.159 bytes	0x298b5443390 0f1bbfe295ae25 dd4c1014b8f6c 76793f3b7b707 744eab7cbc55a	QmZuCsUJ AcvcijgkW asrKwGbcn PYM5WgC jiSAtcTLT RRC8

Delivering files to nodes without the same swarm key is prohibited in this study's private IPFS network. Downloading files from the Public IPFS network and unregistered users can accomplish this in situations used for confidentiality testing. In this instance, the private IPFS network's file to be downloaded has a valid hash value, but the public IPFS network's swarm key differs from the private IPFS network's swarm key. The results of confidentiality testing are reported in Table 9. The test results demonstrate that the proposed SCIS system's private IPFS network can maintain file confidentiality. Only registered nodes for the private IPFS network with the same swarm key get files. While the download attempt from a public IPFS network or an unregistered user was unsuccessful, it was successful from within the private network.

Table 9. Results of Confidentiality Testing

Network	Valid Hash	Testing	Result
Private IPFS Network	QmV4asVwV9 owLkNJe8o7ZP LAiZhfetonxng c4X96TFQ7Db	Download	Successful
Public IPFS Network	QmV4asVwV9 owLkNJe8o7ZP LAiZhfetonxng c4X96TFQ7Db	Download	Failed
Unregistered Users	QmV4asVwV9 owLkNJe8o7ZP LAiZhfetonxng c4X96TFQ7Db	Download	Failed

3.8. Evaluation

In the evaluation stage of the availability, integrity, and confidentiality of files in the SCIS system, several tests were carried out according to the scenarios designed. First, the availability of files in the SCIS System is done by deactivating one of the four nodes in the IPFS private network used by the system. This is done to simulate a system failure or cyber-attack. In the test results, it was found that when one node is deactivated, data can still be accessed through nodes that are still active in the private IPFS network. This shows that a SCIS system using IPFS technology can provide a high level of redundancy and is not dependent on a central point that is vulnerable to attack or system failure and ensures data availability in difficult situations.

DOI: https://doi.org/10.29207/resti.v7i3.4956

Second, file integrity testing is done by uploading the original file and modified file with almost the same content and size into the SCIS system. The test results show that the SCIS system can detect changes made in files, even just a few bytes. This happens because the IPFS that resides in the SCIS system uses hash technology which can ensure the integrity of the information by creating the file's hash and storing the hash into the IPFS. By using this hash, IPFS can ensure that data stored on the network cannot be modified without the knowledge of users on the network.

Finally, file confidentiality testing is carried out by simulating unauthorized access from malicious entities from outside the SCIS system. In this test, downloads are performed from entities that have not registered as members of the IPFS private network in the SCIS system, as well as entities that are on a different network from the IPFS private network but have a valid hash value or match the desired file. However, the test results show that downloads from unregistered entities and entities outside the IPFS private network cannot be performed or fail because entities outside the IPFS private network do not have a common swarm key as members of the private IPFS network so access to download or upload files to the system cannot be performed. Therefore, this test proves that the SCIS system can provide a high level of confidentiality for the data stored in it.

4. Conclusion

This paper describes a simulation of a SCIS system that combines private IPFS and a private Ethereum blockchain. According to the findings of the tests, files are distributed equitably across all nodes in a private IPFS network in which any node can contribute data. SCIS System has answers to problems with a single point of failure, changing data, data confidentiality, and making more data available. Nodes cannot join a private IPFS network without the correct swarm key. Public IPFS networks can't get hold of legitimate files so SCIS systems can guarantee the privacy of information. Moreover, transaction logs are kept as proof of transactions on the private Ethereum blockchain. The Ethereum blockchain is immutable after it has been digitally signed with an Ethereum account, and every participant in it is aware of the precise time and date of every transaction. Finally, the SCIS system can guarantee the confidentiality, availability, and integrity of cybersecurity information exchanged between sectoral organizations using the private Ethereum Blockchain network and private IPFS network, allowing sectoral organizations to share and securely utilizing cybersecurity information.

Suggestions for future research, SCIS systems should incorporate the entire cybersecurity information lifecycle and provide IPFS with a wider network to store large-capacity files.

Acknowledgment

This research received financial support from the Indonesian Ministry of Communication and Informatics

References

[1] A. Lamssaggad, N. Benamar, A. S. Hafid, and M. Msahli, "A Survey on the Current Security Landscape of Intelligent Transportation Systems," *IEEE Access*, vol. 9, no. Vlc, pp. 9180–9208, 2021. https://doi.org/10.1109/ACCESS.2021.3050038.

[2] H. S. Lallie *et al.*, "Cyber security in the age of COVID-19: A timeline and analysis of cyber-crime and cyber-attacks during the pandemic," *Comput Secur*, vol. 105, p. 102248, 2021. https://doi.org/10.1016/j.cose.2021.102248.

[3] S. J. Ee, J. W. Tien Ming, J. S. Yap, and S. C. Y. Lee, "Active and Passive Security Attacks in Wireless Networks and Prevention Techniques," pp. 1–17, 2020. https://doi.org/doi: 10.36227/techrxiv.12972857.v1.

[4] K. Bhat, E. Van Der Kouwe, H. Bos, and C. Giuffrida, "ProbeGuard: Mitigating Probing Attacks Through Reactive Program Transformations," *International Conference on Architectural Support for Programming Languages and Operating Systems - ASPLOS*, pp. 545–558, 2019. https://doi.org/10.1145/3297858.3304073.

[5] A. B. Ajmal, M. A. Shah, C. Maple, M. N. Asghar, and S. U. Islam, "Offensive Security: Towards Proactive Threat Hunting via Adversary Emulation," *IEEE Access*, vol. 9, pp. 126023–126033, 2021. https://doi.org/10.1109/ACCESS.2021.3104260.

[6] N. N. P. Mkuzangwe and Z. C. Khan, "Cyber-Threat Information-Sharing Standards: A Review of Evaluation Literature," *The African Journal of Information and Communication*, no. 25, pp. 1–12, 2020. https://doi.org/10.23962/10539/29191.

[7] Berlilana, T. Noparumpa, A. Ruangkanjanases, T. Hariguna, and Sarmini, "Organization benefit as an outcome of organizational security adoption: The role of cyber security readiness and technology readiness," *Sustainability (Switzerland)*, vol. 13, no. 24, 2021. https://doi.org/10.3390/su132413761.

[8] Y. Li and Q. Liu, "A comprehensive review study of cyber-attacks and cyber security; Emerging trends and recent developments," *Energy Reports*, vol. 7, no. xxxx, pp. 8176–8186, 2021. https://doi.org/10.1016/j.egyr.2021.08.126.

[9] J. Dykstra, L. A. Gordon, M. P. Loeb, and L. Zhou, "The Economics of Sharing Unclassified Cyber Threat Intelligence by Government Agencies and Departments," *Journal of Information Security*, vol. 13, no. 03, pp. 85–100, 2022. https://doi.org/10.4236/jis.2022.133006.

[10] A. Pala and J. Zhuang, "Information sharing in cybersecurity: A review," *Decision Analysis*, vol. 16, no. 3, pp. 172–196, 2019. https://doi.org/10.1287/deca.2018.0387.

[11] S. Pal, M. Hitchens, T. Rabehaja, and S. Mukhopadhyay, "Security requirements for the internet of things: A systematic approach," *Sensors (Switzerland)*, vol. 20, no. 20, pp. 1–34, 2020. https://doi.org/10.3390/s20205897.

[12] M. A. Berawi, M. Sari, F. A. F. Addiani, and N. Madyaningrum, "Developing a Blockchain-based Data Storage System Model to Improve Government Agencies' Organizational Performance," *International Journal of Technology*, vol. 12, no. 5, pp. 1038–1047, 2021, doi: 10.14716/ijtech.v12i5.5237.

DOI: https://doi.org/10.29207/resti.v7i3.4956

[13] M. Hajizadeh, N. Afraz, M. Ruffini, and T. Bauschert, "Collaborative cyber attack defense in SDN networks using blockchain technology," *Proceedings of the 2020 IEEE Conference on Network Softwarization: Bridging the Gap Between AI and Network Softwarization, NetSoft 2020*, no. June, pp. 487–492, 2020. https://doi.org/10.1109/NetSoft48620.2020.9165396.

[14] A. Mallik, A. Ahsan, M. M. Z. Shahadat, and J. C. Tsou, "Man-in-the-middle-attack: Understanding in simple words," *International Journal of Data and Network Science*, vol. 3, no. 2, pp. 77–92, 2019. https://doi.org/10.5267/j.ijdns.2019.1.001.

[15] P. Purwono *et al.*, "Blockchain Technology," vol. 8, no. 2, pp. 199–205, 2022. https://doi.org/10.26555/jiteki.v8i2.24327.

[16] A. G. Gad, D. T. Mosa, L. Abualigah, and A. A. Abohany, "Emerging Trends in Blockchain Technology and Applications: A Review and Outlook," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 9, pp. 6719–6742, 2022. https://doi.org/10.1016/j.jksuci.2022.03.007.

[17] H. Taherdoost, "Smart Contracts in Blockchain Technology: A Critical Review," *Information (Switzerland)*, vol. 14, no. 2, 2023. https://doi.org/10.3390/info14020117.

[18] R. Tas and O. O. Tanriover, "Building A Decentralized Application on the Ethereum Blockchain," *3rd International Symposium on Multidisciplinary Studies and Innovative Technologies, ISMSIT 2019 - Proceedings*, pp. 1–4, 2019. https://doi.org/10.1109/ISMSIT.2019.8932806.

[19] N. Gowda and C. Chakravorty, "Comparative study on cryptocurrency transaction and banking transaction," *Global Transitions Proceedings*, vol. 2, no. 2, pp. 530–534, 2021. https://doi.org/10.1016/j.gltp.2021.08.064.

[20] S. Nzuva, "Smart Contracts Implementation, Applications, Benefits, and Limitations," *Public Policy and Administration Research*, no. October, 2019. https://doi.org/10.7176/ppar/9-9-06.

[21] M. Muneeb, Z. Raza, I. U. Haq, and O. Shafiq, "SmartCon: A Blockchain-Based Framework for Smart Contracts and Transaction Management," *IEEE Access*, vol. 10, pp. 10719–10730, 2022. https://doi.org/10.1109/ACCESS.2021.3135562.

[22] J. E. de Azevedo Sousa *et al.*, "An analysis of the fees and pending time correlation in Ethereum," *International Journal of Network Management*, vol. 31, no. 3, 2021. https://doi.org/10.1002/nem.2113.

[23] T. Ncube, N. Dlodlo, and A. Terzoli, "Private Blockchain Networks: A Solution for Data Privacy," *2020 2nd International Multidisciplinary Information Technology and Engineering Conference, IMITEC 2020*, no. December, 2020. https://doi.org/10.1109/IMITEC50163.2020.9334132.

[24] S. Namane and I. Ben Dhaou, "Blockchain-Based Access Control Techniques for IoT Applications," *Electronics (Switzerland)*, vol. 11, no. 14, pp. 1–29, 2022. https://doi.org/10.3390/electronics11142225.

[25] M. A. Manolache, S. Manolache, and N. Tapus, "Decision Making using the Blockchain Proof of Authority Consensus," *Procedia Comput Sci*, vol. 199, pp. 580–588, 2021. https://doi.org/10.1016/j.procs.2022.01.071.

[26] A. Al-Zoubi, T. Saadeddin, and M. Dmour, "An Ethereum Private Network for Data Management in Blockchain of Things Ecosystem," *International Journal of Online and Biomedical Engineering (iJOE)*, vol. 19, no. 01, pp. 38–58, 2023. https://doi.org/10.3991/ijoe.v19i01.35261.

[27] E. Nyaletey, R. M. Parizi, Q. Zhang, and K. K. R. Choo, "BlockIPFS - Blockchain-enabled interplanetary file system for forensic and trusted data traceability," *Proceedings - 2019 2nd IEEE International Conference on Blockchain, Blockchain 2019*, no. September, pp. 18–25, 2019. https://doi.org/10.1109/Blockchain.2019.00012.

[28] J. Shamdasani, "Decentralized File Storage (Interplanetary File System) using Blockchain," vol. 12, no. 03, pp. 252–256, 2023. https://doi.org/10.1016/j.procs.2023.01.088.

[29] G. Gurung, R. Shah, and D. P. Jaiswal, "Software Development Life Cycle Models-A Comparative Study," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, no. March, pp. 30–37, 2020. https://doi.org/10.32628/cseit206410.

[30] C. Munoz-Ausecha, J. E. G. Gómez, J. Ruiz-Rosero, and G. Ramirez-Gonzalez, "Asset Ownership Transfer and Inventory Using RFID UHF TAGS and Ethereum Blockchain NFTs," *Electronics (Switzerland)*, vol. 12, no. 6, pp. 1–20, 2023. https://doi.org/doi: 10.3390/electronics12061497.

[31] S. K. Panda and S. C. Satapathy, "An Investigation into Smart Contract Deployment on Ethereum Platform Using Web3.js and Solidity Using Blockchain," no. July, pp. 549–561, 2021. https://doi.org/10.1007/978-981-16-0171-2_52.

[32] J. Zheng *et al.*, "An In-Depth Review on Blockchain Simulators for IoT Environments," *Future Internet*, vol. 14, no. 6, pp. 1–22, 2022. https://doi.org/10.3390/fi14060182.

DOI: https://doi.org/10.29207/resti.v7i3.4956