Data Sharing using PDPA-Compliant Blockchain Architecture in Malaysia

Hasventhran Baskaran[1], Salman Yussof[2], Asmidar Abu Bakar[3], Fiza Abdul Rahim[4]
Australian Matriculation Department, Sunway College, Bandar Sunway, Malaysia[1]
Institute of Informatics and Computing in Energy, Universiti Tenaga Nasional, Kajang, Malaysia[2, 3]
Advanced Informatics Department, Razak Faculty of Technology and Informatic, Kuala Lumpur, Malaysia[4]

Abstract—**Data privacy is undoubtedly the biggest concern for the modern society. Data privacy is also becoming a key policy in data protection regulations. Organizations assemble massive amount of personal data of the users for monetary and political purposes. These data can be sold for commercial purpose without the prior knowledge or permission from the respective data owners. This can be mitigated by having blockchain to provide a much-needed transparency. However, blockchain's own transparency becomes its own disadvantage when data owners want to be completely anonymous. Blockchain's transparent nature will be conflicting with non-linkability. Since the data in blockchain is publicly viewable, any personal data or private transactions being processed through blockchain will be exposed to every node in the network. Hence, blockchain implementations also must comply with privacy acts such as Personal Data Protection Act (PDPA) to have privacy by design and by default. Hence, this paper proposes a PDPA-compliant blockchain architecture for data trading that provides complete control of data and anonymity to the users. A prototype is created using various tools to implement the proposed architecture. This study presents anonymous data sharing for users, data access, data delete features to verify the correctness of the proposed architecture.**

Keywords—Blockchain; smart contract; legitimacy; access protocol; retention protocol; data processor; data subject; data user; blockchain regulations

I. Introduction

Sharing data is one of the most basic operations being performed on the Internet. Data can be shared to one or many depending on the need for the operation [1]. Nowadays, data is being considered as a valuable asset due to the growth of Internet of Things (IoT), cloud computing and big data [2]. There are many benefits of data sharing that underlines its importance. For example, data sharing can benefit enterprises, research and also our society [3]. Enterprises can reduce their inefficiencies, increase collaborations and open up new ventures between businesses. Researchers can build more connections and collaboration with other researchers. This will allow them to work on novel research topics rather than repeating existing works. The society can benefit through information shared to increase economic benefits, improved innovations and higher rates of advancement in technology and medical research.

However, there are some issues with data sharing that concerns the parties involved in it. Data integrity and user anonymity are very serious aspects of data sharing that regularly concern the data owners. Data owners might have less trust on data management solutions due to the fear of their personal data being exposed and exploited without their permissions. It is also possible for data sharing to result in the violation of the users' privacy and losing control over the data produced by the users themselves [4].

We chose blockchain as the solution to tackle the issues that rise with data sharing. The users' concerns with data sharing mentioned previously can be handled by blockchain. Blockchain is a distributed ledger technology that was popularized by cryptocurrency applications. It is a ledger that consists of a record of transactions. From information technology perspective, blockchain can be seen as a database. This technology is designed to facilitate distributed transactions without depending on a central point of authority. Multiple users can make changes to the ledger simultaneously. The ledger will be stored in multiple nodes that participates in the blockchain network.

Blockchain technology rose to popularity with the rise of Bitcoin and many cryptocurrencies followed the suit. Cryptocurrencies are a virtual coin that holds a certain value that can be monetized. Users can buy, sell, and trade these cryptocurrencies with blockchain. Blockchain is immutable in nature so it will preserve data integrity. This enables data integrity to be preserved and prevent any false transactions.

Since blockchain is decentralized and transparent, any transactions happening in the network will be publicly available to all users. Blockchain eliminates the need for a trusted third party and ensure the data ownership of the users by enforcing smart contracts. Users do not have to worry about their data being exploited because the usage and the users of data can be defined by the data owners using smart contract. These features give the users an increased control and trust over their data. The core concept can also be used to enable users to trade their data in return for money in Malaysian context. Instead of sending cryptocurrencies through blockchain, users can also trade different types of data through blockchain.

However, some additional components need to be added to a generic blockchain architecture to protect user's data privacy due to blockchain's transparent nature. Users' personal data could be publicly viewable if it is shared on blockchain. Users also should be able to control how much and to whom their data can be exposed to. Blockchain addresses' linkability; also needs to be handled to not expose the identities of the users.

Personal Data Protection Act 2010 (PDPA) in Malaysia is an act that enforces the regulation of the processing of personal data concerning commercial transactions. A bill relating to the PDPA of Act 709 was passed by the Malaysian Parliament on 16th May 2011. Following this, the Personal Data Protection Department was established under the Ministry of Communications and Multimedia Commission (MCMC). PDPA is applicable to any person who processes and collects data for commercial purposes. Hence, this research discusses the current blockchain ecosystem in Malaysia and proposes the first PDPA-compliant blockchain architecture for data trading purposes in Malaysia.

Hence, this paper discusses the current blockchain ecosystem in Malaysia and proposes the first PDPA-compliant blockchain architecture for data trading purpose in Malaysia. This paper contains six sections. Section I is the introduction and explains the need for implementing a user centric data privacy protection for data trading on blockchain platform. Section II provides the relevant works done in relation to data privacy and blockchain. Section III introduces our proposed PDPA-compliant blockchain architecture. Section IV discusses implementation and testing of the proposed blockchain architecture using tools such as Multichain, Solidity, Truffle and PYPSV (Python library). Section V provides the conclusion for this paper.

II. Related Works

This section discusses the definition of data privacy and the types of profitable data associated with it. It also discusses briefly about blockchain technology and its components, data management solutions with blockchain, and the relevance of PDPA to blockchain.

A. Data Privacy

The concern for data privacy is ever increasing among us today, considering the amount of big data generated by our daily life activities. Data privacy is also becoming a key policy in data protection regulations.

We are witnessing massive developments in statistics and computer science that enable computers to analyse and interpret data automatically. These processed data are being used by computers for decision making and knowledge discovery. Nowadays, we can see computers are used for decision making in various sectors. The shift from decision making by humans to automated decision making will raise multiple issues and leave a huge impact.

When thousands or millions of individual data are combined together, they can form digital profiles of people using the particular service. This data can be used for monetary purpose or to shift tides in politics. Some types of data that can be used for these purposes are as follows [5, 8]:

- Age
- Gender
- Address
- Job Title
- Ethnicity
- Religion
- Salary
- Personality traits

While the concept of privacy is very subjective due to the difference in individuals and their cultures, it shares a common theme of having the “right to be alone” [6]. Smith et al. defines privacy as the concern of an individual regarding the access to his/her identifiable personal information [7]. The ISO_IEC_15408-2_2008 standard is referred to identify the data privacy properties. The standard is stating anonymity, psedonymity, unlinkability and unobservability as the required properties.

Anonymity is when the data user or subject who own the data might act without releasing their user identity to others such as users, subjects, or objects. If a subject is anonymously performing an action, another subject will not be able to determine either the identity or even a reference to the identity of the user employing the subject.

Pseudonymity is when a subject can be referred back directly by being related to a reference (alias) held by the Data Controller, or by providing an alias that will be used for processing purposes. The alias could be in the forms such as an account number or smart contract ID if it’s a blockchain implementation. Both pseudonymity and anonymity protect the identity of the user, but in pseudonymity a reference to the user's identity is maintained for accountability or other purposes.

Unlinkability is intended to protect the user identity against the linking of the operations done by the user. Unlinkability requires that different operations cannot be related. For example, the user associated with the operation, or the terminal which initiated the action, or the time the action was executed.

A number of techniques can be applied to implement unobservability. The information might be allocated to a single randomly chosen part of the system/architecture/nodes such that an attacker does not know which part of the system/architecture/nodes should be attacked. An alternative system might distribute the information such that no single part of the system/architecture/nodes has sufficient information that, if circumvented, the privacy of the user would be compromised. It can be achieved by methods such as cryptography and encryption

B. Blockchain Technology

Blockchain is a type of distributed ledger technology that contains the transactions shared by participating parties in the network. Blockchain maintains an immutable record of this transactions with a P2P network where the copy of the ledger is duplicated to all the participating nodes [9]. Every node gets to keep a copy of the ledger. Transactions in blockchain are validated and authorized by these nodes using a consensus algorithm.

Blockchain can be separated into four types, public, private, hybrid and consortium based on usability [10]. There are different trade-offs for each type. There are various consensus algorithms such as Proof of Work, Proof of Stake, and Proof of

Byzantine Fault Tolerance that can be used by blockchain to verify the transactions made [11].

Blockchain also eliminates the need for a trusted third party by providing digital trust through its smart contract [12]. Users are not required to share any personal information to make transactions through blockchain. Transactions are verified by comparing the provided keys as no authority figure is needed. These features enable blockchain to be a good platform to introduce user-centric data management solutions.

C. Data Management Solutions using Blockchain

Since data market places are becoming a common phenomenon, there is a great need for data management solutions that satisfies users. Data management solutions in blockchain are already witnessing increased adoption. Blockchain is a suitable platform for data sharing because of its decentralization architecture and the ability to have a system that eliminates the need for a trusted third-party.

Bajoudah et al. proposed a decentralized marketplace that facilitates IoT data brokering among selected nodes. This will require minimal trust since the smart contract is enforced to carry out the instructions sent by the user and the transaction details are recorded on a public network [13]. They are using Ethereum's public network and smart contract to mediate the interaction among data producers and consumers. This method ensures complete transparency and non- repudiability.

The authors in [14] proposed a data trading framework based on smart contract using machine learning and blockchain. The authors enforce data trading centers to not be able to retain the shared data during the data trading process with their solution. Hence, the framework is designed to eliminate the need for a trusted third party and give complete control to the data producers. An off-chain download mechanism and challenge response mechanism to download the purchased data and to authenticate data owner are incorporated into the framework.

A consortium blockchain-based data trading framework for the Internet of Vehicles (IoV) was proposed by Chen et al. in [15]. They applied an iterative double auction mechanism that induces users to submit bids and decide the amount of data to trade and its price. The proposed framework's algorithm also extracts the hidden information of participants gradually to ensure their privacy is protected.

The authors mentioned above provided great data management solutions with blockchain. However, the PDPA has a set of regulations that has to be complied by those who makes data-centric solutions. Both regulations definitely affect the way previous researchers dealt with data, especially personal information. Hence, these solutions do not comply fully with key principles of PDPA such as the right to be forgotten and the need for a data controller.

Onik et al. has proposed a solution to have a privacy-aware blockchain for personal data sharing and tracking that complies with European Union's General Data Protection Regulation (GDPR) [16]. Any businesses or organizations that deal with EU subjects has to comply with GDPR even if they are outside the EU. GDPR ensures the protection of its citizens' personal data regardless of the territories [19,20]. The proposed system in [16] stores personal data on off-chain storage and successfully tracks the data movement between the parties involved in the data sharing transaction. However, due to the tracking feature, it does not satisfy one of the properties of data privacy which is unlinkability. Hence, the proposed architecture in this paper takes non-linkability as a fundamental feature and build upon it. The architecture aims to provide complete anonymity and control to the data producers/owners.

D. PDPA, Blockchain and Data Trading

Malaysia's PDPA is an Act that regulates the processing of personal data in regards to commercial transactions. It was gazetted in June 2010. The penalty for not complying to PDPA is between RM100,000 to 500,000 and/or between 1 to 3 years imprisonment. The importance of PDPA as provided by the Department of Personal Data Protection [17] is as follows:

- To enhance public confidence and trust with ongoing enforcement
- To avoid and minimize the incidents of data breach
- To increase the efficiency and governance of personal data
- To ensure prudence and integrity in personal data handling

Data Subjects are the owners of the data being uploaded to the blockchain architecture. Data Processors are the third parties that are interested to buy data produced by Data Subjects. Data controller is the authority figure that can verify transactions and help make data trading for data user and Data Subject [18]. There should be at least one Data Controller to enforce the rights of the Data Subjects.

Data controllers must obtain explicit consent from users before making any transfer of the data to the Data Processors [18]. The controllers are also required to have a list of consents given by the Data Subjects. This rule can be applied by getting smart contracts from the data and storing the transaction details in the blockchain.

PDPA also encourages PDPA's Retention and Access Principles mandate that data can be modified or erased to comply with legal requirements as mentioned in Section 34, 35, 36 and 37 of PDPA [18]. This conflicts with blockchain's immutable nature that emphasizes on data integrity. Data cannot be modified or erased from blockchain.

However, it can be solved by storing the data on off-chain storage instead on the blockchain. The PDPA sections mentioned earlier also can be enforced by having functions in the smart contract that can access or delete data required by the Data Subjects. PDPA is also applicable only when the data breach happens in Malaysia. This can be enforced by having an off-chain storage in Malaysia.

III. Proposed Blockchain Architecture

A PDPA compliant blockchain architecture is proposed based on the discussion in our previous paper [21] where we studied the gaps between PDPA and blockchain and also the

PDPA-compliant features, and PDPA [18] as shown in Table I.

TABLE I. PDPA COMPLIANT FEATURES

Feature	PDPA	Description
Smart Contracts	Right to erasure and modification, consent to collect data	Both regulations emphasizes on enabling the users to erase or modify the data shared by them. Both regulations requires for the consent to be collected from the users before collecting their data.
Stealth Address [22]	Not Associated	By using stealth address, only the sender and receiver can determine where a data was sent. They allow and require the sender to create random one-time addresses for every transaction on behalf of the recipient. This provides complete anonymity.
Off-chain storage [23-25]	Right to erasure, Territorial limitation	Data on blockchain cannot be erased. However, this can be solved by using off-chain storage. The PDPA can only be enforced if the data storage or breach happens in Malaysia. A off-chain storage in Malaysia will ensure immediate enforcement of PDPA.
Data Controllers	Having data controller is recommended	Data Controllers are needed to enforce or process the data on behalf of the Data Subjects. A minimum of 2 Data Controllers are needed for a Data Subject.

Table I provides the explanation for the PDPA features implemented in our proposed architecture. This section provides the detailed description of the blockchain based PDPA-compliant Data Sharing architecture. This study uses a permissioned blockchain where three stakeholders, Data Subject, Data Controllers and Data Processors form the blockchain network.

The key terms such as Data Subject, Data Controllers and Data Processor are used here and explained below:

- Data Subjects are the data producers.
- Data Controllers are the legal entities that determines the purpose of the data processing and enforces the smart contracts on behalf of the Data Subjects.
- Data Processors are the third-party companies that process data on behalf of the Data Controller.

These terms are from the PDPA as the blockchain architecture is designed to be PDPA compliant. The architecture is designed to have the implementation of key principles of PDPA as the core features. Our definition of data in this proposed architecture is subjective. It can be any type of data that the users want to share. The data type can vary from personal information, bank transactions, energy consumption data, health information and etc.

The architecture enables Data Subjects to share data produced by them to a Data Controller in return for a monetary or point-based reward. The reward received by the Data Subjects can be stored in their preferred digital or cryptocurrency wallet depending on the type of reward given to them. Data Processors can purchase these datasets from the Data Controller for data processing purpose.

Fig. 1 shows the proposed blockchain architecture. This architecture implements stealth address, an access protocol and a retention protocol. It comprises of three layers, user layer, system management layer and storage layer. The green dotted lines indicate the interactions of the nodes with blockchain. The solid lines indicates the interaction between the stakeholders. Stealth Address (SA), Access Protocol (AP) and Retention Protocol (RP) are used for the interaction between nodes. AP and RP are explained in sub-section *2* of this section and the implementation is shown in sub-section B under System Implementation and Testing. Components used in the proposed blockchain architecture are described below:

- Node: Data Subjects, Data Controllers and Data Processors.
- Block: A new block is generated and added to the existing blockchain each time data sharing happens successfully among the nodes.
- Block header: A block header stores hash of the previous block, transaction time, transaction ID and data encoding style.
- Transaction data: Storing of smart contract ID and hash of the shared data.
- Consensus algorithm: Round-robin mechanism provided by Multichain.

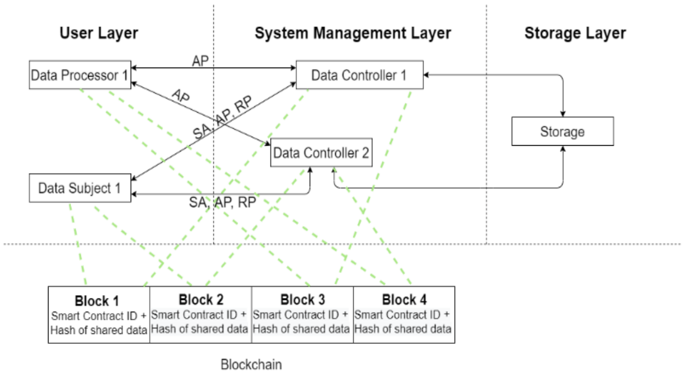


Fig. 1. Proposed blockchain architecture.

1) User layer: Each time Data Subjects want to share some data, they can choose the amount and the types of data to be shared. The system then creates a smart contract with their preferred data types. Then a one-time stealth address is created for users to send their smart contract to the blockchain and Data Processor. Users' smart contract ID and transaction details will be stored in blockchain. The data shared by them will be stored in an off-chain storage. Data Processors exist in this layer to purchase Data Subject's data from the Data Controllers.

2) System management layer: In this layer, data controller enables Data Subjects to claim reward for the data they shared by providing the smart contract ID related to the dataset. Access protocol (AP) facilitates the data exchange between Data Subject, Data Processor and data user. Users send their smart contract ID linked to the dataset they want access to and data controller retrieves the dataset based on it. Retention protocol (RP) is very crucial in exercising Data Subject's rights to erase their data by providing the smart contract ID linked to the particular dataset. Retention protocol is only available for Data Subjects. Both AP and RP are enforced by data controller in this layer.

3) Storage layer: An off-chain storage is used to store the data as it enables PDPA to be enforced should any data breach happens. Once, the user has shared the data to the Data Processor, the data will be sent to the off-chain storage and the transaction details are stored in the blockchain. Since the data is stored in off-chain storage, it is easier to apply AP and RP to access or delete the user's data. Off-chain data is erasable and the remainder of the hash of the shared data will be useless and unidentifiable. This can be ensured by cross-checking the hashes generated after the modification and deletion. This makes the storage layer to comply with PDPA's principle to enable users to modify and erase their data.

4) Creation of blocks: Any transaction in the blockchain will only be added to a block after the creation of the genesis block. The architecture shown in Fig. 1 assumes the genesis block is already created during the instantiation of the blockchain. Then, it shows the block creations and the transactions related to them. The contents and the interaction between the nodes during the block creation are explained below.

- Block1: Data Subject1 shares data to Data Controller1. Block is created after consensus is achieved from the Data Subject1 and Data Controller1. The smart contract ID and the hash of the shared data is stored in the block.

- Block2: Data Subject1 shares data to Data Controller2. Block is created after consensus is achieved from the Data Subject1 and Data Controller2. The smart contract ID and the hash of the shared data is stored in the block.

- Block3: Data Processor1 retrieves data from Data Controller1. Block is created after consensus is achieved from the Data Processor1 and Data Controller1. The smart contract ID and the hash of the shared data is stored in the block.

- Block4: Data Processor1 retrieves data from Data Controller2. Block is created after consensus is achieved from the Data Processor1 and Data Controller2. The smart contract ID and the hash of the shared data is stored in the block.

Stealth Address (SA), Access Protocol (AP) and Retention Protocol (RP) are used for the interaction between nodes. The execution of AP and RP are explained below.

Access Protocol can be used in three different methods. The first two different ways can be used by Data Subject only, where the Data Subject can trigger the AP to create a new dataset or access already shared data. The last method is for Data User to buy dataset from Data Controller. Table II describes the components needed to complete the Access and Retention protocol.

TABLE II. PROTOCOL COMPONENTS FOR AP AND RP

DSi	**Data Subject is the Data Producer / Owner**
DCi	Data Controller determines the purpose of the data processing and enforces the smart contracts on behalf of the Data Subjects.
DPi	Data Processors are the third-party companies that buys data from Data Controller
BC	Blockchain
OCS	Off-chain storage
MSG	Data sent by Data Subjects (e.g: duration and the types of data to be shared)
MSGhash	Hash of the data sent by the Data Subject
DDC	Data sent by Data Controller
SCi_{ID}	Smart Contract ID
$PSCi_{ID}$	Previous Smart Contract ID
Ti_{ID}	Transaction ID
RD	Request for data
PR	Points based reward
DR	Data Price Rate
P	Payment
E	Notice of data erasure

a) Access protocol for sending new dataset:

- DSi creates a smart contract to send a new dataset, MSG. The smart contract is being sent to DCi with SCi_{ID}, Ti_{ID}, and MSG. The SCi_{ID} is stored by DSi for future reference.

- DCi stores the SCi_{ID} Ti_{ID}, and MSGhash in BC for auditing purpose.

- MSG and the associated SCi_{ID} is stored in OCS for future processing, access or deletion.

- DCi pays PR for the MSG shared by the DSi.

Fig. 2 shows the first method of AP to be used by Data Subject to send a new dataset to the Data Controller.

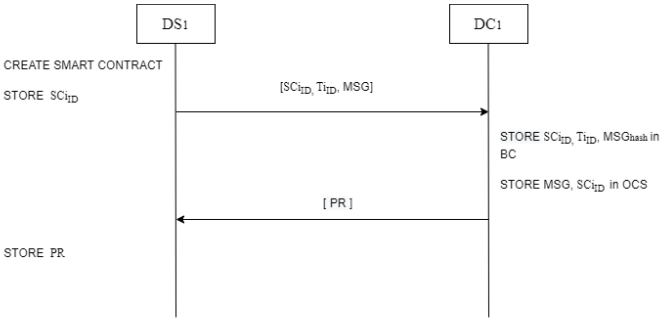

DS1
DC1
CREATE SMART CONTRACT
STORE SCi_{ID}
[SCi_{ID}, Ti_{ID}, MSG]
STORE SCi_{ID}, Ti_{ID}, MSG_{hash} in BC
STORE MSG, SCi_{ID} in OCS
[PR]
STORE PR

Fig. 2. Data subject sending data using AP.

b) Access Protocol for requesting previously shared data:

- DSi creates a smart contract to request for previously shared dataset, MSG. The smart contract is being sent to DCi with SCi_{ID}, Ti_{ID}, and MSG that contains $PSCi_{ID}$. The SCi_{ID} is stored by DSi for future reference.
- DCi checks the match for $PSCi_{ID}$ in OCS to look if matches with any data.
- If there is any match with $PSCi_{ID}$, the MSG associated to it is retrieved.
- The SCi_{ID} and Ti_{ID} is stored in for the current transaction is stored in BC for auditing purpose.
- DCi sends DDC that contains the previously shared MSG to the DSi.

Fig. 3 shows the first method of AP to be used by Data Subject to send a new dataset to the Data Controller.

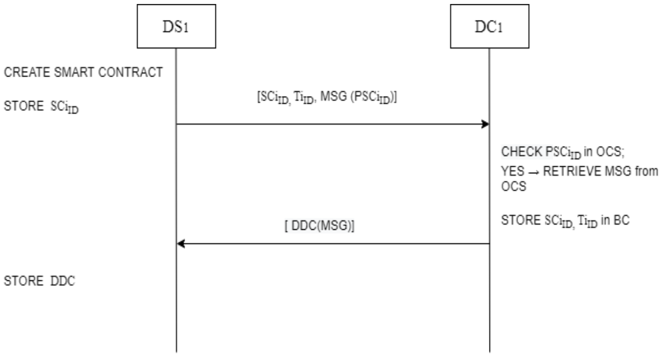

DS1
DC1
CREATE SMART CONTRACT
STORE SCi_{ID}
[SCi_{ID}, Ti_{ID}, MSG ($PSCi_{ID}$)]
CHECK $PSCi_{ID}$ in OCS;
YES → RETRIEVE MSG from OCS
STORE SCi_{ID}, Ti_{ID} in BC
[DDC(MSG)]
STORE DDC

Fig. 3. Data subject using AP to access shared data.

AP is limited in terms of functionality for Data Processor, as shown in Fig. 4.

c) Access Protocol for data purchase:

- DPi creates a smart contract to request for some data. The smart contract is sent to DCi with SCi_{ID}, Ti_{ID}, and DR. The SCi_{ID} is stored by DPi for future reference.
- The SCi_{ID} and Ti_{ID} is stored in for the current transaction in BC for auditing purpose.
- The DCi issues the payment of DR for the data requested by the DPi.
- The DPi then pays the DR to the DPi.
- The DPi DDC from the OCS and sends it to DPi.

Fig. 4 shows the method of AP used by Data Processor to buy a dataset from the Data Controller.

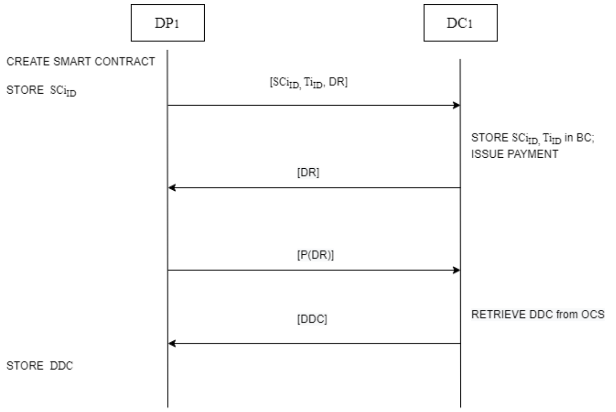


Fig. 4. Data processor requesting data using AP.

d) Retention Protocol:

- DSi creates a smart contract to request of deletion for previously shared dataset, MSG. The smart contract is being sent to DCi with SCi_{ID}, Ti_{ID}, and MSG that contains $PSCi_{ID}$. The SCi_{ID} is stored by DSi for future reference.
- DCi checks the match for $PSCi_{ID}$ in OCS to look if matches with any data.
- If there is any match with $PSCi_{ID}$, the MSG associated to it is erased.
- The SCi_{ID} and Ti_{ID} is stored in for the current transaction is stored in BC for auditing purpose.
- DCi sends E to DSi to complete the transaction.

Fig. 5 shows how RP is used by Data Subject to erase previously shared dataset to the Data Controller.

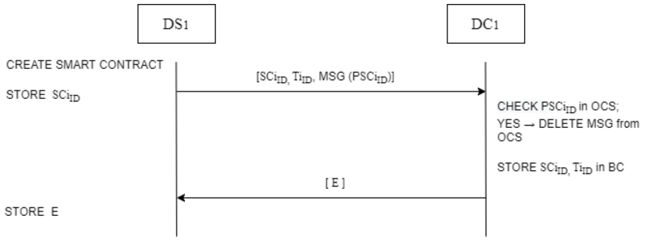

DS1
DC1
CREATE SMART CONTRACT
STORE SCi_{ID}
[SCi_{ID}, Ti_{ID}, MSG ($PSCi_{ID}$)]
CHECK $PSCi_{ID}$ in OCS;
YES → DELETE MSG from OCS
STORE SCi_{ID}, Ti_{ID} in BC
[E]
STORE E

Fig. 5. Data subject requesting the data to be deleted using RP.

IV. Implementation and Testing

This section explains the implementation and testing of the proposed architecture. In the first sub-section, case study for the architecture is discussed. In the following subsection, system implementation and testing of the proposed architecture is explained.

A. Case Study: Data Management Scenario

This case study test is done by referring to a similar case study implemented by the authors in [16]. A use case scenario is described with three different situations to further explain the

PDPA-compliant blockchain architecture. The three situations explained here are: Data Subject-controller data sharing scenario, Data Processor-controller data sharing scenario and data deletion scenario.

a) Data subject-controller data sharing scenario: Every time the Data Subjects send a set of data to the Data Processor, the transaction details and smart contract ID will be stored in the blockchain and in their local storage. For any set of data, it will be linked with the smart contract's ID. The data controller separates the data into data shared by user, hash value generated for the shared data, transaction ID and smart contract ID. A combination of smart contract ID and data shared by user is stored in the off-chain storage. The combination of generated hash value and smart contract ID are stored in a new block and added to the current blockchain.

An appropriate amount of reward for the shared data will be sent to the Data Subject's temporary address created by stealth address mechanism. Then, the Data Subject can transfer the reward to their digital or cryptocurrency wallet. The transfer of the reward to the user's personal wallet will not be recorded on the blockchain as it happens internally on the end device.

When a Data Subject sends a request with AP to data controller to get a copy of a specific set of data, the data controller will verify and grant the request by the Data Subject. Data controller proceeds to check for the data by using the smart contract ID and its matching data in the off-chain storage. If a match is found, the data controller will send a copy of the requested data by retrieving it from the off-chain storage. Transaction details will be stored in the blockchain for auditing purpose.

2) Data processor-controller data sharing scenario: A Data Processor sends a request with AP to Data Controller to get a copy of a specific set of data from the off-chain storage. Data controller will verify and grant the request. Then, the data controller will send a copy of the requested data by retrieving it from the off-chain storage. Transaction details will be stored in the blockchain for auditing purpose. When Data Processor executes the access protocol, they need to pay the Data Controller corresponding to the amount of data they need access to. The payment could be in fiat currency.

3) Data deletion scenario: Data Subject sends a request to data controller with relevant Smart Contract ID to erase their data. Data controller will verify and grant the request. After the verification, the data controller searches for any data associated with the Smart Contract ID provided by the user in the off-chain storage. Any data found with matching Smart Contract ID will be erased and the transaction details are stored in blockchain.

B. System Implementation and Testing

This sub-section presents a pilot implementation scenario of the PDPA-compliant blockchain architecture with various tools. The tools used here are Multichain (for network setup with off-chain storage), Remix IDE and Solidity (for writing smart contract), Truffle suite (to test the smart contract) and *pyspv* library from python (to build stealth address functionality).

1) Node and network setup: A private network with one Data Subject, two controllers and one Data Processor is set up on the blockchain platform with Multichain 2.1.2 [26] as shown in Fig. 6. We have implemented the prototype of the architecture on four computers. The details are as follows:

- Data Controller 1: Windows 10 64-bit, i5-9400F @ 2.90GHZ, Acer Predator Orion 3000.
- Data Controller 2: Windows 10 64-bit, i5-8300H @ 2.30GHZ, Acer Nitro 5.
- Data Subject: Windows 10 64-bit, i7-4600U @ 2.10GHZ, HP Elitebook 840.
- Data Processor: Windows 10 64-bit, A12-9720P @2.70GHZ, HP Laptop 15-bw0xx.

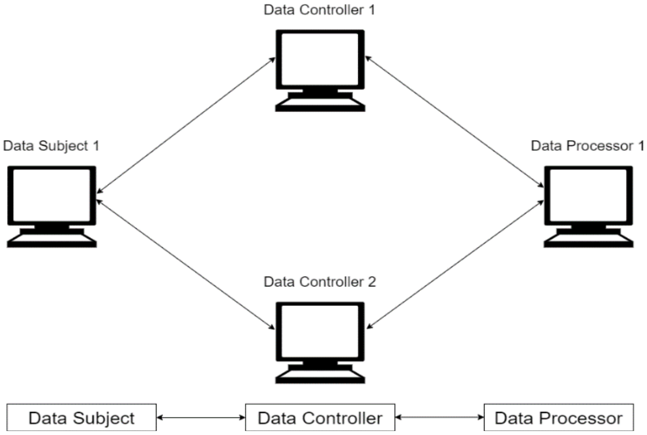
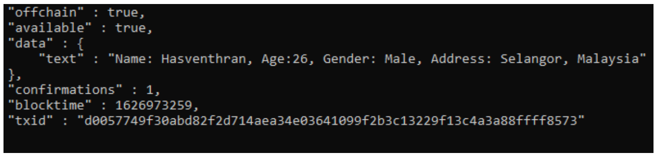


Fig. 6. Experimental network setup.

Multichain is mainly selected due to its off-chain storing mechanism. The built-in off-chain feature enables easy management of data among the nodes. Multichain also helps to add the hash of the off-chain data to the existing block.

Since this is a private network, the consensus only requires minimum difficulties to be achieved. No nonce is used and the consensus is achieved with a round robin mechanism since the node identities are known in a private blockchain. Masking of the node identity will be explained in subsection 3. Multichain is used here for its off-chain storage and easy blockchain deployment capability.

An example of data stored in the off-chain storage is shown in Fig. 7. This is retrieved by the Data Processor1 node after getting the permission from the Data Controller1 to subscribe to the data stream that contains this dataset. The status of off-chain section is shown as true to indicate that the data is being stored in the off-chain storage.



```
"offchain" : true,
"available" : true,
"data" : {
    "text" : "Name: Hasventhran, Age:26, Gender: Male, Address: Selangor, Malaysia"
},
"confirmations" : 1,
"blocktime" : 1626973259,
"txid" : "d0057749f30abd82f2d714aea34e03641099f2b3c13229f13c4a3a88ffff8573"
```

Fig. 7. Data stored in off-chain storage.

2) *Smart contract:*

The smart contract has four functions that helps AP and RP to be applied in the blockchain architecture. This contract was written in Solidity language with Remix IDE. The contract was tested with Truffle Suite on Geth's Ropsten test network. A few researchers have already used Truffle to test their smart contracts [27 - 31].

Algorithm I: Pseudocode for Data Sharing Smart Contract

```
contract DataSharing{
/*Defining variables and their types*/
int ContractId
string memory name
int age
string memory gender
string memory address
int expiry_date

User[] internal users;
uint public currentID equal to 1010;
RETURN currentID

FUNCTION create_data(string name, int age, string gender,string user_address,
int expiry_date){
Set name = name in name
Set age = age in age
Set gender = gender in gender
Set user_address = user_address in user_address
Set expiry_date = expiry_date in expiry_date
Increment currentID by one when this function is called
RETURN currentID
}

FUNCTION search_data(int ContractId) return (name, age, string
gender,user_address, int expiry_date){
Call find_data and pass ContractId to ContractId
Set i = ContractId
RETURN name = name in users[i]
RETURN age = age in users[i]
RETURN gender = gender in users[i]
RETURN user_address = user_address in users[i]
RETURN expiry_date = expiry_date in users[i]

}

FUNCTION delete_data(uint id) {
Call find_data and pass ContractId to id
Set i = id
delete users[i]
}

FUNCTION find_data(uint ContractId) {
Get the value of ContractId
FOR i = 1 to user's length step 1 DO
IF users[i] in ContractId equal to ContractId
RETURN i
ELSE
PRINT "Smart Contract ID does not exist!"
ENDIF
ENDFOR
}
```

The pseudocode above reflects the smart contract written to facilitate the data sharing in the proposed architecture. We referred to the template written by authors in [32] to write the pseudocode for our proposed smart contract. The four functions written in the contract are *create_data*, *search_data*, *delete_data* and *find_data*. There are seven variables named *ContractId*, *name*, *age*, *gender*, *address*, *expiry_date* and *currentID*. An array named *users* is created to contain the instances of the structure, *User*.

In the function *create*, users can fill in the data they want to share. For testing purpose, four personal information such as name, age, gender and address are chosen to be shared by the user. The expiry date of the data is also set by the users (in the format of DDMMYY) to prevent the data from being kept longer than necessary. If users only wish to share their name, they can fill in the remaining fields with "NIL" or "0". Once the user creates their data, the variable *currentID* is incremented by 1.

In the function search_data, index of the structure User that we need is retrieved. User enters their index/smart contract ID and passes it to the function find_data. A for-loop statement is then used to find if the smart contract ID entered by the user exists in record. If there is no relevant entry, an error message is displayed and cancels the current execution. This function is called as the Access Protocol in the architecture.

In the function delete_data, users enter their smart contract ID. The function takes a single argument and calls the function find_data to get the index of the structure in the users array. If the relevant index/smart contract ID is found, the data associated to the index is deleted.

Algorithm II: Truffle test case for smart contract

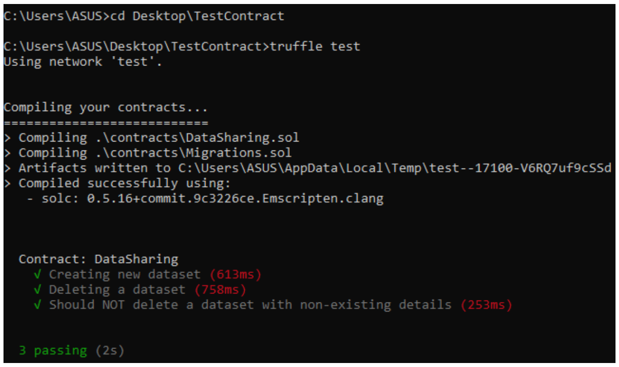
```
const DataSharing = artifacts.require('DataSharing.sol');

contract('DataSharing',() => {
        let datasharing = null;
        before(async() => {
                datasharing = await DataSharing.deployed();
        });

        it('Creating new dataset', async () => {
                await datasharing.create('Hasventhran', 26, 'Male',
'Malaysia', 110721);
                const user_data = await
datasharing.search_data(1010);
                assert(user_data[0].toNumber() === 1010);
                assert(user_data[1] === 'Hasventhran');
                assert(user_data[2].toNumber() === 26);
                assert(user_data[3] === 'Male');
                assert(user_data[4] === 'Malaysia'
                assert(user_data[5] === '110721');
        });


        it('Deleting a dataset', async () => {
                await datasharing.delete_data(1010);
                try {
                        await datasharing.search_data(1010);
                } catch(e) {
                        assert(e.message.includes('Smart
Contract ID does not exist!'));
                        return;
                }
                assert(false);
        });

        it('Should NOT delete a dataset with non-existing details', async
() => {
                try{
                        await datasharing.delete_data(1010);
                } catch(e){
                        assert(e.message.includes('Smart
Contract ID does not exist!'));
                        return;
                }
                assert(false);
        });
```

A test case was written in Javascript to test the smart contract's correctness in the architecture. Three public functions of the smart contract, *create_data*, *search_data* and *delete_data* were included in the test case as shown above. The function *find_data* is not included as it is an internal/private function of the smart contract to access the smart contract ID assigned to the user. *search_data* is a function that depends on the value passed by *find_data*. Successful test of *search_data* ensures the passing of *find_data*.

In the first *it* function in the test case, *create* is tested by adding smart contract ID and five different values according to the data types defined in the smart contract. The *search_data* function is also defined in this function as it can be tested along with the *create_data* and *delete_data* functions. Successful test of these two functions ensures the passing of *search_data*.

The *delete_data* function is separated into two *it* functions. The first function tests whether the dataset is correctly deleted after obtaining the smart contractID from the search_data function. If the smart contract ID does not exist, it should be able to display an error message. The second *it* function checks for the smart contract's correctness in not deleting data with non-existing details.

The contract is tested on Ropsten network of Geth with the truffle command, *truffle test*. As shown in Fig. 8, all three *it* functions of the test has successfully passed the test. Hence, the correctness of the smart contract is ensured. Hence, it can be deployed in the architecture to enforce AP and RP.



```
C:\Users\ASUS>cd Desktop\TestContract

C:\Users\ASUS\Desktop\TestContract>truffle test
Using network 'test'.


Compiling your contracts...
===========================
> Compiling .\contracts\DataSharing.sol
> Compiling .\contracts\Migrations.sol
> Artifacts written to C:\Users\ASUS\AppData\Local\Temp\test--17100-V6RQ7uf9cSSd
> Compiled successfully using:
   - solc: 0.5.16+commit.9c3226ce.Emscripten.clang



  Contract: DataSharing
    √ Creating new dataset (613ms)
    √ Deleting a dataset (758ms)
    √ Should NOT delete a dataset with non-existing details (253ms)


  3 passing (2s)
```

Fig. 8. Successful test of smart contract.

3) Stealth address: A python script is written to test stealth address. This python script uses the pyspv library to run the stealth address functions. This script is tested on a simulated blockchain and it is explained below. Data Controller is mentioned as Data User in the code sections.

```
def main () :
    Data_User_key=pyspv.keys.PrivateKey.create_new()

Data_User_public_key=Data_User_key.get_public_key(
True)
print("Data_User'spublic ke
y=",Data_User_public_key.as_hex())
```

Code Snippet 1 Creating Data Controller's public key

Python's *pyspv* library is used to test stealth address functions. Data Controller's public and private key is created through the code shown in the code snippet 1.

```
Data_Subject_key = pyspv.keys.PrivateKey.create_new()
Data_Subject_public_key=Data_Subject_key.get_public_key(True)
Data_Subject_shared_secret_point=Data_User_public_key.multiply(
Data_Subject_key.as_int())
print("Ephemeral key (Data_User needs this to redeem) = ",
Data_Subject_public_key.as_hex())
```

Code Snippet 2 Data Subject creating new key for Data Controller

The same method applies for the Data Subject in the section of code below. The Data Subject's public and private key is also created and shown in code snippet 2. The Data Subject wants to pay/send data to the Data Controller. So, the Data Controller gives the Data Subject his/her public key. Then, the Data Subject creates a new key and multiplies the Data Controller's public key with his/her private key and sends her public key to the Data Controller as shown in the code in code snippet 2.

```
import hashlib
hasher = hashlib.sha256()
hasher.update(Data_Subject_shared_secret_point.pubkey)
shared_secret = hasher.digest()
print("Shared secret ="", pyspv.bytes_to_hexstring(shared_secret,
reverse=False))
```

Code Snippet 3 Hashing the shared secret

The next task is to hash the shared secret. Python's *hashlib* library is used to hash the shared secret with SHA256 algorithm. This is done by the code in code snippet 3.

```
new_Data_User_public_key=Data_User_public_key.add_constant(int.fro
m_bytes(shared_secret, ''big))
print("New_Data_User Public Key = ",
new_Data_User_public_key.as_hex())
print("New_Data_User=",new_Data_User_public_key.as_address(pypsv.
Bitcoin))
```

Code Snippet 4 Adding shared secret to Data Controller's public key

After hashing the shared secret, it is added to the Data Controller's public key. This will be used later to compute the Data Controller's private key. This is done by the code in code snippet 4.

```
Data_User_shared_secret_point=Data_Subject_public_key.multiply(Data
_User_key.as_int())
```

Code Snippet 5 Computing the private key to Data Controller

```
hasher = hashlib.sha256()
hasher.update(Data_User_shared_secret_point.pubkey)
shared_secret_by_Data_User = hasher.digest()
print("Data_User figured out the shared secret =",
shared_Secret_by_Data_User == shared_secret)
```

Code Snippet 6 Producing the shared secret by hashing private key

In order to compute the private key to the new Data Controller public key, the Data Controller has work to do as shown in code snippet 5. First, the Data Controller multiplies the ephemeral key produced by the Data Subject by his private key. Then, the shared key is produced by the Data Controller by hashing the private key as shown in code snippet 6.

```
new_Data_User_key=Data_User_key.add_constant(int.from_bytes(sh
ared_secret_by_Data_User, ‘big’))
new_Data_User_computed_public_key=new_Data_User_key.get_pub
lic_key(True)
print(“Data_User figured out the correct private key
=”,new_Data_User_computed_public_key.pubkey==new_Data_User
_public_key.pubkey)
```

Code Snippet 7 Adding the Shared Secret to the Private Key

Finally, Data Controller adds the shared secret to his private key as shown in code snippet 7. This works because, given Q = dG, then (d+n)G=dG+nG=Q+nG and the Data Subject computed Q+nG above and sent data to the Data Controller, but now the private key d+n is known.

V. Conclusion

This research aimed to produce a PDPA-compliant user-centric privacy preserving features for blockchain based data sharing architecture. Blockchain was chosen to be the platform to develop the data sharing architecture, due to its inherent privacy properties and also flexibility of implementing other features on top of it. Based on the gaps identified by literature review, features such as private blockchain, use of data controllers, stealth address, smart contracts (with access and retention protocols), and off-chain storage are identified to produce the proposed architecture with user control being given importance. The results indicate that the implemented features fulfil the required PDPA regulations.

The proposed blockchain architecture is developed mainly to comply with PDPA in Malaysia. The architecture is appropriate to be followed for Malaysian context. It is acceptable to use the proposed architecture to develop any data management solutions using blockchain in Malaysia. However, the proposed architecture may not be suitable to be used in a general manner globally due to different data protection regulations all around the world.

The proposed blockchain architecture is developed by combining existing privacy preserving and user-centric components. Moreover, the proposed architecture can be used by blockchain developers to develop data management solutions without worrying about the potential breach of privacy. This is because the proposed architecture adheres to the “privacy by design” principle.

This research proposed a user-centric privacy preserving blockchain architecture for data sharing according to existing data protection regulations. Hence, to better understand the effectiveness of the architecture, future studies can implement the architecture for various type of data sharing solutions across multiple fields. These new solutions derived from the proposed architecture can also be modified according to compliance with each country’s data privacy protection regulations.

Acknowledgment

This work is funded by Tenaga Nasional Berhad Seed Fund (U-TD-RD-19-24) in collaboration with TNB Distribution Network. We would like to thank UNITEN R&D Sdn. Bhd. for their role in fund management. The publication of this paper is funded by the International NEC Energy Transition Grant (202202001ETG).

References

[1] Pandey, V. and Kulkarni, U., Effective data sharing with forward security: Identity based ring signature using different algorithms. 2017 International Conference on Intelligent Computing and Control (I2C2), 2017.

[2] Jin, H., Luo, Y., Li, P. and Mathew, J., A Review of Secure and Privacy-Preserving Medical Data Sharing. IEEE Access, 7, pp.61656-61669, 2019.

[3] Data Republic, The importance of data sharing, https://www.datarepublic.com/resources/resources-guides/the-importance-of-data-sharing-for-all-organizations, Accessed 18th July 2021.

[4] OECD iLibrary, Risks and challenges of data access and sharing, Enhancing Access to and Sharing of Data : Reconciling Risks and Benefits for Data Re-use across Societies, https://www.oecd-ilibrary.org/sites/15c62f9c-en/index.html?itemId=/content/component/15c62f9c-en, Accessed 7th August 2021.

[5] M. Kosinski, D. Stillwell and T Graepel, "Private traits and attributes are predictable from digital records of human behavior", Proc. Nat. Acad. Sci. USA, vol. 110, pp. 5802-5805, 2013.

[6] [20] S. D. Warren and L. D. Brandeis, "The right to privacy", Harvard Law Rev., vol. 4, no. 5, pp. 193-220, 1890.

[7] H. J. Smith, T. Dinev and H. Xu, "Information privacy research: An interdisciplinary review", MIS Quart., vol. 35, no. 4, pp. 989-1015, 2011.

[8] Wickramaarachchi, W., Alhaj, Y. and Gunesekera, A., Effective Privacy-Preserving Iris Recognition. 2019 IEEE 4th International Conference on Image, Vision and Computing (ICIVC),2019.

[9] Belen Saglam, R., Aslan, C., Li, S., Dickson, L. and Pogrebna, G., A Data-Driven Analysis of Blockchain Systems' Public Online Communications on GDPR. 2020 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS), 2020.

[10] Parizo, C., 2021. What are the 4 different types of blockchain technology?, https://searchcio.techtarget.com/feature/What-are-the-4-different-types-of-blockchain-technology , Accessed 25th July 2021.

[11] GeeksforGeek, Consensus Algorithms in Blockchain, https://www.geeksforgeeks.org/consensus-algorithms-in-blockchain/, Accessed 9th May 2021.

[12] Baskaran, H., Yussof, S. and Rahim, F., A Survey on Privacy Concerns in Blockchain Applications and Current Blockchain Solutions to Preserve Data Privacy, 2020.

[13] Bajoudah, S., Dong, C. and Missier, P., Toward a Decentralized, Trust-Less Marketplace for Brokered IoT Data Trading Using Blockchain. 2019 IEEE International Conference on Blockchain (Blockchain), 2019.

[14] Xiong, W. and Xiong, L., Smart Contract Based Data Trading Mode Using Blockchain and Machine Learning. IEEE Access, 7, pp.102331-102344, 2019.

[15] Chen, C., Wu, J., Lin, H., Chen, W. and Zheng, Z., A Secure and Efficient Blockchain-Based Data Trading Approach for Internet of Vehicles. IEEE Transactions on Vehicular Technology, 68(9), pp.9110-9121, 2019.

[16] Onik, M., Kim, C., Lee, N. and Yang, J., Privacy-aware blockchain for personal data sharing and tracking. Open Computer Science, 9(1), pp.80-91, 2019.

[17] Department of Personal Data Protection, https://www.pdp.gov.my/jpdpv2/assets/2020/01/Introduction-to-Personal-Data-Protection-in-Malaysia.PDF, Accessed 24th July 2021.

[18] Personal Data Protection Act 2010 (Act 709). Malaysia: Parliament of Malaysia, 2010.

[19] Edwards, J., 6 business benefits of data protection and GDPR compliance, https://searchdatabackup.techtarget.com/tip/6-business-benefits-of-data-protection-and-GDPR-compliance, Accessed 17th July 2021.

[20] General Data Protection Regulation (GDPR), General Data Protection Regulation (GDPR) – Official Legal Text. https://gdpr-info.eu, Accessed 25th April 2021.

[21] Baskaran, H., Yussof, S., Rahim, F. and Bakar, A., Blockchain and the Personal Data Protection Act 2010 (PDPA) in Malaysia. 2020 8th International Conference on Information Technology and Multimedia (ICIMU), 2020.

[22] Patrick, What are Stealth Addresses?, Mycryptopedia. Available at: https://www.mycryptopedia.com/everything-need-know-stealth-addresses/, Accessed 19th April 2021.

[23] Kumar, R., Marchang, N. and Tripathi, R., Distributed Off-Chain Storage of Patient Diagnostic Reports in Healthcare System Using IPFS and Blockchain. 2020 International Conference on COMmunication Systems & NETworkS (COMSNETS), 2020.

[24] Li, H. and Han, D., EduRSS: A Blockchain-Based Educational Records Secure Storage and Sharing Scheme. IEEE Access, 7, pp.179273-179289, 2019.

[25] Kumar, R. and Tripathi, R., A Secure and Distributed Framework for sharing COVID-19 patient Reports using Consortium Blockchain and IPFS. 2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC), 2020.

[26] Multichain.com. 2021. MultiChain for Developers | MultiChain, https://www.multichain.com/developers/, Accessed 15th May 2021.

[27] Wang, S., Li, D., Zhang, Y. and Chen, J., Smart Contract-Based Product Traceability System in the Supply Chain Scenario. IEEE Access, 7, pp.115122-115133, 2019.

[28] Gupta, R., Shukla, A. and Tanwar, S., AaYusH: A Smart Contract-Based Telesurgery System for Healthcare 4.0. 2020 IEEE International Conference on Communications Workshops (ICC Workshops), 2020.

[29] Patidar, K. and Jain, S., Decentralized E-Voting Portal Using Blockchain. 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2019.

[30] Ranganthan, V., Dantu, R., Paul, A., Mears, P. and Morozov, K., A Decentralized Marketplace Application on the Ethereum Blockchain. 2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC), 2018.

[31] BouSaba, C. and Anderson, E., Degree Validation Application Using Solidity and Ethereum Blockchain. 2019 SoutheastCon, 2019.

[32] Kirit, N. and Sarkar, P., EscrowChain: Leveraging Ethereum Blockchain as Escrow in Real Estate, ResearchGate, 2017.