Integration of Cross-Enterprise Document Sharing (XDS.b) with Blockchain Technology

Petnathean Julled

Faculty of ICT, Mahidol University

Nakhon Pathom, Thailand

petnathean.jul@student.mahidol.ac.th

Assadarat Khurat

Faculty of ICT, Mahidol University

Nakhon Pathom, Thailand

assadarat.khu@mahidol.ac.th

Pattanasak Mongkolwat

Faculty of ICT, Mahidol University

Nakhon Pathom, Thailand

pattanasak.mon@mahidol.ac.th

**Corresponding author:**

Assadarat Khurat

999 Phuttamonthon 4 Road, Salaya , Nakhon Pathom 73170 THAILAND

assadarat.khu@mahidol.ac.th

Tel: +66(0)816152243

Abstract—Healthcare information sharing and interoperability between healthcare organizations are important factors to healthcare quality and safety since a patient may require medical services from different healthcare providers. Integrating Healthcare Enterprise (IHE) provides Cross-Enterprise Document Sharing-b (XDS.b) profile that allows the organizations to share health documents between each others. However, security is a big challenge inhibit successful data sharing such as data integrity, availablity, and privacy. In addition, no specific security implementations were endorsed for XDS.b. Blockchain technology can be applied to prevent some of cyber threat issues facing health information sharing. A novel method using Blockchain technology to ensure health information integrity and availability is implemented, demonstrated and freely available, allowing health document sharing through decentralized network while addressing cyber-security issues through unique characteristics of Blockchain technology.

Keywords—health information sharing, information security, blockchain, Smart Contract, IHE

Integration of Cross-Enterprise Document Sharing (XDS.b) with Blockchain Technology

line 1: 1st Given Name Surname   
line 2: *dept. name of organization   
(of Affiliation)*  
line 3: *name of organization   
(of Affiliation)*line 4: City, Country  
line 5: email address or ORCID

line 1: 2nd Given Name Surname  
line 2: *dept. name of organization   
(of Affiliation)*  
line 3: *name of organization   
(of Affiliation)*line 4: City, Country  
line 5: email address or ORCID

line 1: 3rd Given Name Surname  
line 2: *dept. name of organization   
(of Affiliation)*  
line 3: *name of organization   
(of Affiliation)*line 4: City, Country  
line 5: email address or ORCID

*Abstract—Healthcare information sharing and interoperability between healthcare organizations are important factors to healthcare quality and safety since a patient may require medical services from different healthcare providers. Integrating Healthcare Enterprise (IHE) provides Cross-Enterprise Document Sharing-b (XDS.b) profile that allows the organizations to share health documents between each others. However, security is a big challenge inhibit successful data sharing such as data integrity, availablity, and privacy. In addition, no specific security implementations were endorsed for XDS.b. Blockchain technology can be applied to prevent some of cyber threat issues facing health information sharing. A novel method using Blockchain technology to ensure health information integrity and availability is implemented, demonstrated and freely available, allowing health document sharing through decentralized network while addressing cyber-security issues through unique characteristics of Blockchain technology.*

Keywords—health information sharing, information security, blockchain, Smart Contract, IHE

# Introduction

The increasing demands for secure, reliable, accurate and instance access to health information among Healthcare Information Exchange network (HIE) have significant impacts on outcomes of patient care and operational efficiency within the participating organizations. Sharing health information with organizations that do not adhere to security standards and practices could expose patients and organizations in HIE to unacceptable risks. The diminishing risk-reward ratio associated with sharing patient information between healthcare institutions may be unfavorable if done incorrectly. These interoperation problems cause a huge decrease in efficiency in healthcare operations and result in a lower quality of healthcare service [1–3].

Integrating Healthcare Enterprise (IHE) [4] uses healthcare standards to guide healthcare interoperability and it has provided the Cross-Enterprise Document Sharing Profile (XDS.b Profile) [5] profiles to address the issue of health information sharing between multiple organizations. The profile provides guidelines essential for system implementers to follow.

The cyber-security threats have been a critical issue threatening the healthcare domain since the first known ransomware attack in 1989 when AIDS researchers effected by a ransomware [6]. Since then, the number of data breaches in healthcare domain has been increasing every year from 2012 to 2021 [7–13]. The breaches raise a concern about data integrity and accessibility to healthcare systems. Since majority of healthcare information systems are centralized systems, when one system goes down, all the connected systems are also affected.

Blockchain technology has been used to ensure sanctity of data because the immutable characteristics obtained from blockchain's cryptographic components, consensus mechanism, and decentralization system will help ensure its availability [14], [15] and information sharing between different enterprises. In this work, we propose an integration of XDS.b profile with Blockchain technology to enable cross-enterprise health document sharing.

# Related Work

## A Blockchain-Based Approach to Health Information Exchange Networks

Peterson et al. (2015) [16] applied Blockchain and HL7 FHIR as gateways, which allow members of the network to access health information from each other while ensuring distribution of accessibility within the network by publishing those gateways to Blockchain. The work introduced the idea of "Proof of Interoperability", as opposed to cost inefficient Proof of Work, and of "Secure Index" allowing encrypted content searching and using FHIR Profile for data discovery in the blockchain network while making health information sharing possible. However, the work did not explicitly mention about how Blockchain technology can be beneficial for health information sharing in term of cybersecurity. We adopted the idea of using Blockchain as a medium for health information exchange networks. Additionally, several suggestions provided in the work also function as guidelines for our Blockchain design, which enables compatibility with the healthcare information environment.

## “MedRec” Prototype for Electronic Health Records and Medical Research Data

MedRec [17] utilized Ethereum’s Smart Contract to store metadata about record ownership, permissions, and data integrity representing existing medical records that are kept in individual nodes on the network. It built on the work of Zyskind et al. (2015) [18], they utilized some cryptographical characteristics of Blockchain to provide an accessible "bread crumb trail" that allows data users to trace back medical history to improve operational efficiency. MedRec helps reduce barriers to effective data sharing by addressing interoperability issues caused by economic incentives that encourage "health information blocking," which is a practice in health IT that interferes with access, exchange, or use of electronic health information [3]. We adopted the Ethereum and its Smart Contract in our Blockchain implementation.

## Blockchain-based electronic healthcare record system for healthcare 4.0 applications

Tanwar et al. (2020) [19] provided a survey of how blockchain was incorporated into electronic health records from 2016 to 2019. The survey also proposed a blockchain-based access control system for electronic health records by using IBM Hyperledger fabric as a medium for health information exchange within organizations. Like MedRec, their concept emphasizes improving access control and audit trail over information exchange between each unit of the healthcare service workflow. The work sets another example of effectively implementing blockchain Smart Contract for the possibility of healthcare blockchain.

# Background Knowledge

## Cross-Enterprise Document Sharing-b (XDS.b) Profile

Cross-Enterprise Document Sharing Profile-b (XDS.b) [5] allows healthcare providers who are members of a health document sharing network (called "XDS Affinity Domain") to share and discover health documents stored in the document repositories of other institutions using a central registry. As shown in Fig. 1, healthcare providers, acting as Document repository, can publish (ITI-42) metadata of their health documents to store at the central registry (Document registry). Other healthcare providers, acting as Document consumer, can search (ITI-18) for a required health document from the central registry. Once the preferred document is found, the Document consumer can systematically request (ITI-43) the document from its source directly. The red-dash line shows the scope of the XDS.b components focusing in this work.

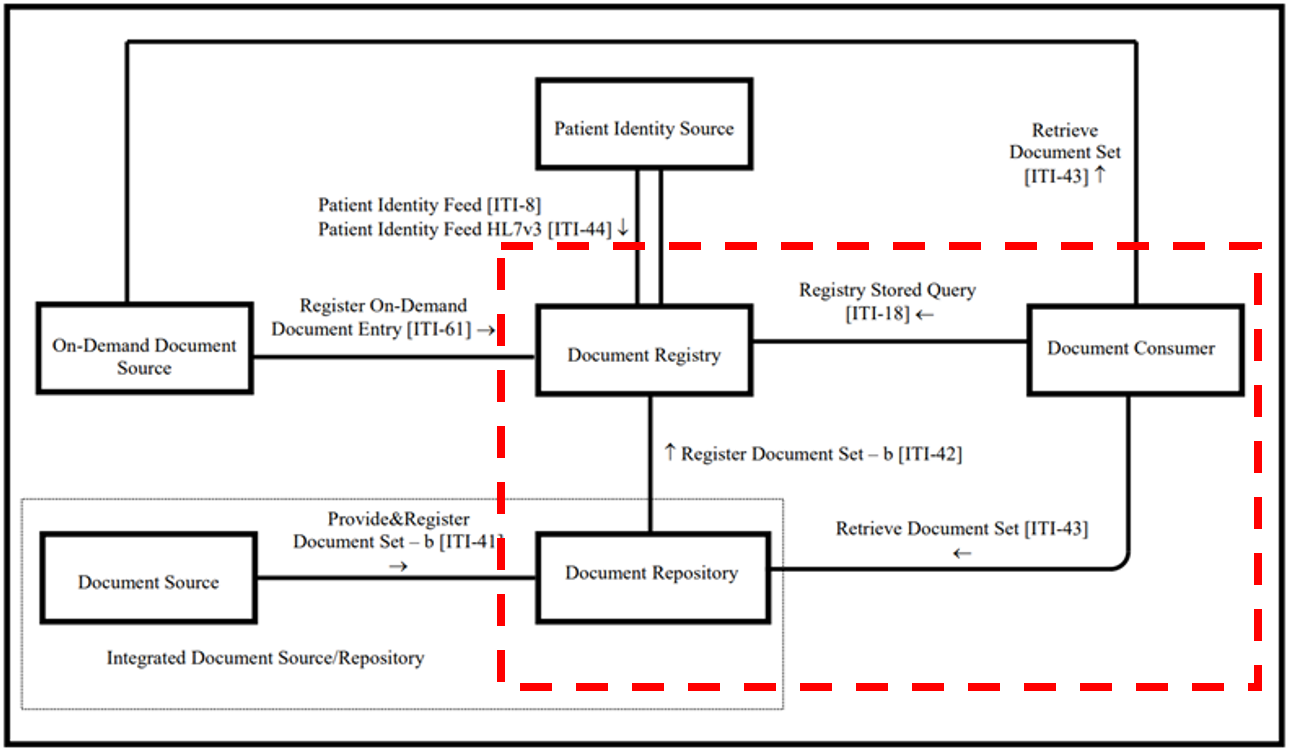


Fig. 1. Cross-Enterprise Document Sharing – Set b [15]  
(Red-dash line) The scope of XDS essential for understanding this work

The IHE IT infrastructure standards tend to refer to each identity, machine, or system as an "actor". IT infrastructure (ITI) transactions which are in XML message, are used to communicate between actors. Each transaction consists of metadata attributes formatted according to the IHE ITI framework standard. For the ITI-18, we focus only two search types i.e., “FindDocument” and “GetDocument” where the former will query a list of documents that match the condition and the latter will query a specific document.

## Blockchain Technology

Blockchain is a technology that applies cryptographic techniques e.g., hashing to locally ensure integrity of data enabling decentralization [20]. It actually creates a chain of block that every blockchain member can check if data in the block is correct. Blockchain can be categorized into two main types: permissionless or public blockchain and permissioned or private blockchain. The former allows everyone to access blockchain network and data freely while the latter requires permission before anyone can join the network and access data. Blockchain employs consensus mechanism to ensure integrity of data i.e., to have accuracy and the same set of data, through agreement and consensus of network members. Three major types of consensus mechanisms are Proof of Work (PoW), Proof of Stake (PoS), and Proof of Authority (PoA). Unlike PoW that requires lots of computational resources, PoS requires blockchain members (called nodes) to stake of certain currency, and PoA requires nodes to vote.

In the current state of the development of blockchain technology, a piece of data contained within a blockchain is commonly called a “transaction”. When a set of transactions are stored together within a blockchain, they form a series and become a “blockchain ledger”. Each blockchain member has their own copy of the blockchain ledger, independent from the others.

Each blockchain platform has its own method and protocol for exchanging data between its members (nodes). These protocols are made so that a single member of a blockchain can send data to all other members at the same time at   
the peer-to-peer level. This allows all members of blockchain network to send and receive transactions with other members and also allows them to check the progression of each other during the block validation process determined by consensus mechanism.

Most of the blockchain platforms have the similar core processes . The block validation process of each transaction block is divided into cycles of processing. Each cycle results in one block being accepted by all nodes and added into the blockchain ledger. The block validation cycle starts with each node broadcasting its transactions to all other nodes in the blockchain network via a peer-to-peer protocol. The node that is designated or assigned as a block validator (miner in PoW) gathers all of the broadcast transactions. At a set time or when a certain condition is met, the validator (or miner) node hashes all the transactions it has collected into a block and suggests it to all the other nodes. Then it depends on each consensus mechanism to determine whose block or which block will be accepted by the entire blockchain network. After the accepted block is declared, all nodes then add the accepted block into their blockchain ledger before beginning the new block validation cycle.

## Ethereum and Smart Contract

Ethereum [21] is one of the most famous open-source blockchain. Besides cryptocurrency, Ethereum is used in various applications such as decentralized finance, games, and gambling [22]. It applies concept of “Smart Contract” [23] allowing developers to integrate their small-size code or snippet of logic into Blockchain. This code allows us to execute any computations such as transfer digital coin to other account. Its Smart Contract relies on JavaScript-like language called ‘Solidity’. The main version of the Ethereum network relies on PoW and PoS for consensus mechanisms, and the blockchain itself is a public type. In addition, the Ethereum developer community has made "forked" versions of the network. One of them is "Quorum" [24] designed to support other consensus mechanisms besides PoS and PoW, and to make it easier for blockchain developers to create private or permissioned chains. Our experiment will use Quorum as the blockchain platform for implementation.

# Proposed Method

## Conceptual Design

Patient data cannot be put directly into Blockchain as it will become persistent due to its replica is distributed all over the network. Instead of risking confidentiality of healthcare data, we propose not to publish healthcare data in blockchain rather publish only metadata of the document. This fits well with the XDS.b profile as it also shares only metadata of the health document. The difference is that the Document Registry now is not the only one who stores the metadata but all the blockchain nodes. In addition, since only metadata are published, healthcare data are at its source. In case, the source becomes unavailable due to e.g., ransomware attack, this makes the healthcare data inaccessible. To mitigate this problem, we propose to allow organizations that have downloaded healthcare data from original sources to act as an additional data backup by providing additional access points using URLs for their downloaded data.

## Blockchain Design

Considering health document sharing scenario using XDS Affinity domain network, members of the network should not be able to freely join without verification. Besides that, each member should agree on sharing of its health documents to other members while not exposing information shared within the network to the outside. Based on this situation, we decide to use permissioned blockchain where joining members must be accepted by the network and have a proper data sharing agreement.

Since we use permissioned blockchain, we choose PoA as the consensus mechanism. This is because it requires less computational resources and no stake. One of the widely adopted subtypes of PoA is Istanbul Byzantine Fault Tolerance (IBFT) [25] that requires some randomly selected blockchain members to verify the integrity of metadata through voting system. We will use IBFT in our blockchain design.

## Integrating Blockchain with XDS.b Profile

To integrate Blockchain into XDS.b Profile, we propose to employ XDS Document Registry to acts as a blockchain node. Thus, the healthcare metadata are shared and searched via blockchain network. These tasks are done through Smart Contract.

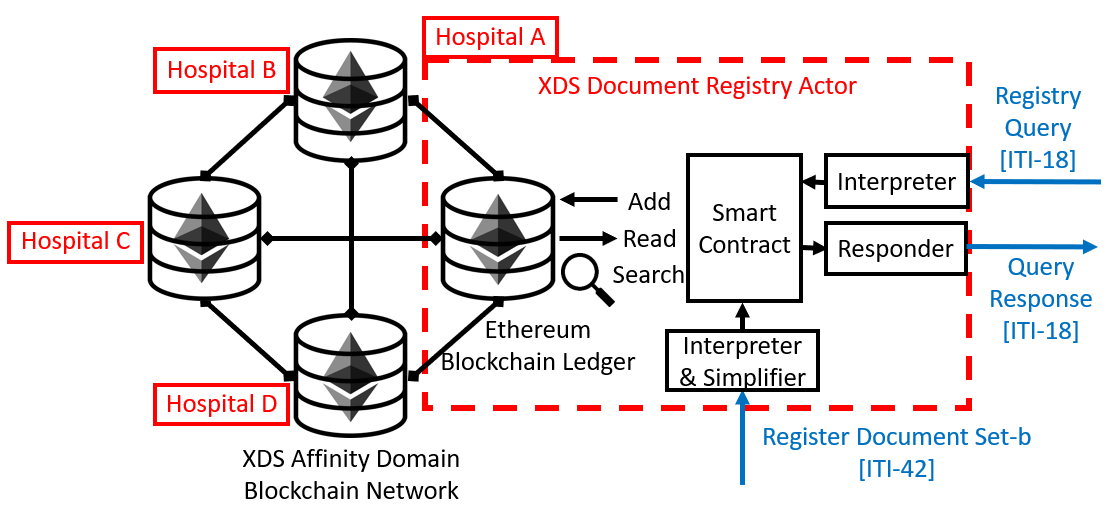


Fig. 2. Integrating Blockchain into XDS.b Profile

As shown in Fig. 2, each Blockchain node will receive ITI-42 transactions from their local Document Repository Actor as usual. The Document Registry Actor then interpretes and transforms the ITI-42 transaction into a form that is compatible with Smart Contract. The Smart Contract then adds metadata and publishes it into the blockchain ledger. The healthcare document query via ITI-18 transactions from local Document Consumer will be interpreted and interacted with Smart Contract. This query contains a list of search keywords chosen by users. Smart Contract will read the keywords and search data within the Blockchain ledger. After the search function is done, the Document Registry Actor creates a response message following the ITI-18 transaction format and sends the search result back.

## Design Functions of Document Registry Actor

As mentioned in previous section, Document Registry Actor registers health document metadata set into the Blockchain ledger and provides location of document queried based on its metadata. In order to implement these two functions, we introduce two Smart Contract functions described as follows.

The first function is "document registering" where the Document Repository Actor submits metadata of newly shared document to the Document Registry Actor. Then these metadata attributes will be fed into a Smart Contract. Due to the limitations of Smart Contracts, it is not practical to store an object that takes a lot of space. This is because large transactions will negatively affect the blockchain network. It adds unnecessary workload to validator nodes and contributes to the accumulation of an overly large blockchain. That means anything stored in the blockchain ledger must be simple and as small as possible. Therefore, we simplify the object variable containing metadata attributes into a simpler form that takes up as little space as possible. Then the simplified variable is assigned into single Smart Contract function transaction.

For the ”document searching” function, generally Blockchain is not designed for searching since what users want to know is the latest data e.g., current account balance not the previous one. Therefore, current blockchain system only keeps unique identifier (using hash value) of the latest data. A straightforward idea could be to keep all of unique identifiers of each document instead.. However, this can be complicated and cumbersome task since each blockchain node has to establish and maintain an off-chain database by itself to record the mapping between unique identifiers and published documents. Another idea is………Therefore, we propose to use the programmability properties of Smart Contracts to solve this issue. เริ่มจากการ design smart contract แล้วใส่ doc id By attaching one more integer variable to the variable containing the metadata attributes set, we can use the Smart Contract to identify itself with this variable. That means the document registration function also automatically assigns each set of metadata with an identification number to differentiate each set of metadata belong to each health document.

The second function is the "document searching" function, where the Document Registry Actor receives search keywords set by the user from the Document Consumer Actor via the document query (ITI-18) transaction and starts using these search keywords as a reference to find the right document. As mentioned above, we assigned an integer variable to each smart contract to act as an identifier. These identifiers will then be used by the search function to search iteratively through all transactions. While iterating, the search function only needs to call the variable containing metadata attributes to get metadata attribute values without the need to make any changes to the transaction. The search function then compares the metadata attributes' values with the search keywords. Depending on which search type is used, the search process can be different. For example, if “FindDocument” is used the search will check all transactions but for “GetDocument”, the search will stop once it found match. Then the metadata attributes of the document in the search result will be added to the ITI-18 response transaction and sent from the Document Registry Actor back to the Document Consumer Actor. Otherwise, the Document Registry Actor will send an ITI-18 response transaction with no results found instead.

# Implementation and Results

This section describes our implementation to demonstrate the concept mentioned previously. The last subsection shows our experimental results.

## Machine Setup

This implementation runs on a virtual machine with Ubuntu Linux OS version 20.04.4 LTS 64-bit, The host machine is Alienware 17 R5 which got 32.0 GB of RAM, an Intel Core i9 CPU at 2.90 GHz, and LLVM 12.0.0 256-bit GPU. We utilize Quorum [26] to run IBFT blockchain nodes in our experiment. The computational resources were divided into 8 Ubuntu virtual machines which would act as Quorum blockchain nodes. Each VM got 2 GB of RAM and 80 GB of storage space just to be able to sustain Quorum, local program and avoid memory swapping effect with the host machine. JavaScript is the main programming language as we utilize NodeJS [27] framework to operate all off-chain program.

อธิบายเพิ่มเกี่ยวกับ scenario ของเครื่องที่ใช้ เช่น เราทำ VM 8 ตัว สเปคแต่ละตัวมีอะไร

อธิบายวิธีทำการทดลอง เช่น ทดสอบ Docregis, GetDoc, FindDoc แล้วทำ 1by1, all

การทดลองแบ่งออกเป็น 2 ส่วนใหญ่ คือ ทดสอบ performance เวลาที่มีการเรียกใช้ function Document Register โดย XDS Document Repository Actor และ เวลาที่มีการเรียกใช้ function Document Query โดย XDS Document Consumer Actor โดยในแต่ละประเภทแบ่งย่อยได้เป็น

1. Document Register ลงทะเบียนข้อมูลแฟ้มข้อมูลเหตุการณ์การรักษา (ด้วย metadata attributes)

1.1 ทดสอบเรียกใช้การลงทะเบียน document โดย node เดียว ทีละ node ทดสอบไล่จาก node ที่ 1 ไปจนถึง node ที่ 8 ไม่พร้อมกัน โดยสลับไปมาระหว่างลงทะเบียน document sample หมายเลข 01 กับ document sample หมายเลข 10 จับระยะเวลาที่ node ใช้ในการ deploy smartcontract จนสำเร็จ

1.2 ทดสอบเรียกใช้การลงทะเบียน document ด้วยทุก node พร้อมกันทั้ง 8 node โดยสลับไปมาระหว่าง document sample 01 และ document sample 10 จับระยะเวลาที่ node ใช้ในการ deploy smartcontract จนสำเร็จ

2 Document Query เรียกดูข้อมูลแฟ้มเหตุการณ์การรักษาที่ถูกลงทะเบียนไว้

2.1.1 ทำการ search ค้นหา document (Find Document) ด้วยจำนวน search keywords (metadata attributes value) เฉพาะประเภท essential (minimum ตามหลักเกณฑ์ระบุโดย IHE) ได้แก่ รหัสประจำตัวผู้ป่วย (Patient ID) และสถานะของแฟ้มข้อมูล โดยทำการ search ด้วย node ทีละ node ไม่พร้อมกัน ทั้งนี้ การ search ประเภทนี้จะตอบกลับเพียง list ของ document unique id ของ document ทั้งหมดที่ match เท่านั้น

2.1.2 ทำการ search ค้นหา document (Find Document) ด้วยจำนวน search keywords (metadata attributes value) เฉพาะประเภท essential (minimum ตามหลักเกณฑ์ระบุโดย IHE) ได้แก่ รหัสประจำตัวผู้ป่วย (Patient ID) และสถานะของแฟ้มข้อมูล โดยทำการ search ด้วย node พร้อมกันทั้ง 8 node

2.2.1 ทำการ search ค้นหา document (Find Document) ด้วยจำนวน search keywords (metadata attributes value) ที่สามารถระบุได้ทั้งหมด ซึ่งรวมทั้ง essential keywords และ optional keywords มาด้วยกัน (maximum ตามหลักเกณฑ์ระบุโดย IHE) ได้แก่ รหัสประจำตัวผู้ป่วย (Patient ID), สถานะของแฟ้มข้อมูล และรหัสหน่วยงาน ส่วนงาน และองค์ประกอบข้อมูลต่าง ๆ ที่สัมพันธ์กับการักษาและโครงสร้างของสถานพยาบาล โดยทำการ search ด้วย node ทีละ node ไม่พร้อมกัน

2.2.2 ทำการ search ค้นหา document (Find Document) ด้วยจำนวน search keywords (metadata attributes value) ที่สามารถระบุได้ทั้งหมด ซึ่งรวมทั้ง essential keywords และ optional keywords มาด้วยกัน (maximum ตามหลักเกณฑ์ระบุโดย IHE) ได้แก่ รหัสประจำตัวผู้ป่วย (Patient ID), สถานะของแฟ้มข้อมูล และรหัสหน่วยงาน ส่วนงาน และองค์ประกอบข้อมูลต่าง ๆ ที่สัมพันธ์กับการักษาและโครงสร้างของสถานพยาบาล โดยทำการ search ด้วย node พร้อมกันทั้ง 8 node

2.3.1 ทำการ search เรียกข้อมูลเต็มของ document (Get Document) ด้วย search keyword ที่เป็น document unique id ตัวเดียว โดยทำการ search ด้วย node ทีละ node ไม่พร้อมกัน

2.3.2 ทำการ search เรียกข้อมูลเต็มของ document (Get Document) ด้วย search keyword ที่เป็น document unique id ตัวเดียว โดยทำการ search ด้วย node พร้อมกันทั้ง 8 node

## XDS Document Repository Actor

Algorithm 1 shows the algorithm of the XDS Document Repository Actor program. Since we cannot find a real document in ITI-41 transaction which is the actual input of the Repository Actor and we only found the document in ITI-42 transaction [27], our program obtains this ITI-42 transaction as input directly. We have implemented a command-line interface to receive a file name of the transaction given by users. The program reads the file to check that it is really ITI-42 and then sends it to the Registry Actor via TCP/IP protocol. Finally, the response will be sent to the users.

|  |  |
| --- | --- |
| **Algorithm 1** XDS Document Repository Actor | |
| **Input:** Sample document  **Output:** Response indicating success or fail to register document  **Initialization:** The user starts the program to imitate the XDS Document Repository's behavior when it receives new documents | |
| 1: | **Start** |
| 2: | **Assign** FileName = user-input |
| 3: | **if** selected Filename == ITI-42 Transaction XML Header |
| 4: | **Send** created ITI-42 to XDS Document Registry Actor |
| 5: | Wait for response |
| 6: | **if** Ack Received = TRUE **then** |
| 7: | Show response and succession marker |
| 8: | **Stop program** |
| 9: | **else** |
| 10: | Notify user to check input and try again |
| 11: | **Stop program** |

## XDS Document Consumer Actor

Algorithm 2 is the algorithm of the XDS Document Consumer Actor program. The program first asks for user-input i.e., search type for the document query and search keywords that are specific to the chosen search type . For user-friendly, we implement the program to receive user inputs as command-line user interface providing choices for users to choose. After that, the program generates an ITI-18 transaction from the inputs before sending it to the Registry Actor and waits for the response. Finally, the search result will be extracted and shown to the user.

| **Algorithm 2** XDS Document Consumer Actor | |
| --- | --- |
| **Input:** User-defined metadata attributes value (search keywords)  **Output:** Search result  **Initialization:** The user starts the XDS Document Consumer Actor program. | |
| 1: | **Start** |
| 2: | **Assign** SearchType = user-selected choice |
| 3: | **for** (Loop until All Keywords Specified) **do** |
| 4: | **Assign** metadata attribute type = user-selected choice |
| 5: | **Assign** SearchKeywords[i] = metadata attributes type  and user-input value |
| 6: | **end** |
| 7: | **Create** new ITI-18 Transaction (SearchType) |
| 8: | **Add** search keywords into ITI-18 (SearchKeywords) |
| 9: | **Send** ITI-18 to XDS Document Registry Actor |
| 10: | Wait for response |
| 11: | **if** Query response received **then** |
| 12: | Extract (query response) |
| 13: | Read search result |
| 14: | Show search result to user |
| 15: | **Stop program** |

## XDS Document Registry Actor

Algorithm 3 is the algorithm of the XDS Document Registry Actor program. The program is always on standby and waits for incoming XML message transactions, either ITI-42 or ITI-18.

Upon receiving the ITI-42 transaction, the program then converts the XML message into JSON using xml2js JavaScript module [28]. Then, the program assorts the JSON object, converts it into the Smart Contract compatible format, and adds the converted object and an integer number used for search function into the newly created Smart Contract and also assigns an integer number for the document to be used in the search function. Then, the program starts putting the Smart Contract into the blockchain ledger through transaction broadcast and block validation. If it fails to publish, the program will retry until successful. When the publishing process is done, the program returns to the standby state and waits for further incoming transactions. Note that the only case where publishing can fail is when the blockchain network is in the middle of a block validation process and cannot properly receive broadcasted transactions. This means that after a few tries, the program will eventually publish smart contracts into the blockchain ledger.

| **Algorithm 3** XDS Document Registry Actor | |
| --- | --- |
| **Input:** ITI-42 Transaction  **Output:** Document registering response (Ack)  **Initialization:** Start new thread upon receiving ITI-42 transaction | |
| 1: | **if** Transaction header is ITI-42 **then** |
| 2: | **Assign** RawXML = Read ITI-42 Transaction (in XML) |
| 3: | **Assign** metadataJSON = Convert RawXML to JSON |
| 4: | **end** |
| 5: | **if** metadataJSON is valid **then** |
| 6: | **Assign** metadataSorted = Assort metadataJSON |
| 7: | **for** Loop through all attributes in metadataSorted **do** |
| 8: | metadataValue [i] = metadataSorted[i] to String |
| 9: | **End** |
| 10: | **Add** metadataValue and integer ID to Smart Contract |
| 11: | **End** |
| 12: | **if** Smart Contract is valid **then** |
| 13: | Publish Smart Contract into Blockchain |
| 14: | Wait for Smart Contract publish response |
| 15: | **if** Smart Contract failed to publish **then** |
| 16: | Try again until successful |
| 17: | **Stop thread and wait for next transaction** |
| 18: | **End** |
| **Input:** ITI-18 Transaction  **Output:** ITI-18 Response Transaction (search result)  **Initialization:** Start new thread upon receiving ITI-18 transaction | |
| 19: | **if** Transaction header is ITI-18 **then** |
| 20: | **Assign** RawXML = Read ITI-18 Transaction (in XML) |
| 21: | **Assign** QueryJSON = Convert RawXML to JSON |
| 22: | **Assign** SearchType = QueryJSON [0] |
| 23: | **for** Loop through all available search keywords **do** |
| 24: | **Assign** Search Keywords [i] = QueryJSON [i + 1] |
| 25: | **end** |
| 26: | **for** Loop through published Smart Contract **do** |
| 27: | **if** metadata in Smart Contract = search keywords **then** |
| 28: | **if** SearchType = FindDocument |
| 29: | searchResult = matched document list (referenced by metadata attributes: entryUUID or uniqueId) |
| 30: | **else** (SearchType = GetDocument) |
| 31: | searchResult = metadataValue in Smart Contract |
| 32: | **end** |
| 33: | **if** reached the end with no match **then** |
| 34: | searchResult = no result found |
| 35: | **end** |
| 36: | **end** |
| 37: | **Create** new ITI-18 Response Transaction as JSON |
| 38: | **Add** searchResult to ITI-18 Response JSON |
| 39: | **Convert** ITI-18 Response JSON to XML |
| 40: | **Send** ITI-18 Response to XDS Consumer Actor |
| 41: | **Stop thread and wait for next transaction** |

Upon receiving the ITI-18 transaction, the program converts the XML message into JSON and retrieves the search type and all search keywords from JSON. At lines 28–35, the program runs a search operation through all available Smart Contracts based on search keywords to find matching documents. The search type determines the pattern of search operation. The “FindDocument” search type will search through all documents in blockchain while the “GetDocument” will stop the search once the matching document is found. Then the search result is added to the ITI-18 response transaction. Before being sent to the Consumer Actor, the ITI-18 response transaction is changed from JSON to XML format. When the process is done, the program returns to the standby state and waits for further incoming transactions.

|  |  |
| --- | --- |
| **Algorithm 4** Smart Contract | |
| **Input:** Assign integer number (ID) and metadataValue  **Output:** Blockchain transaction receipt signify succession of publishing data into Blockchain ledger  **Initialization:** Smart Contract was invoked by the XDS Document Registry Actor with Data Store function | |
| 1: | **Start** |
| 2: | Spawning new transaction instance with ID |
| 3: | StoreValue (ID) = Write (metadataValue) |
| 4: | **Stop** |
| **Input:** integer number (ID)  **Output:** metadata attributes value stored in the Smart Contract  **Initialization:** Smart Contract was invoked by the XDS Document Registry Actor with Data Read function with ID specified by the search function | |
| 5: | **Start** |
| 6: | **return** StoreValue (ID) |
| 7: | **Stop** |

We design Smart Contracts to store string values and return the stored values when they are called. As shown in Algorithm 4 when Smart Contract is called to store data, it spawns a new blockchain transaction containing a new data storage instance that will be used for storing new document metadata attributes. This newly spawned transaction will automatically go through the block validation processes of the blockchain network in line 14 of Algorithm 3. Once it passes the validation process, it will be recorded in the blockchain ledger. When the search program calls on reading data, the Document Registry actor simply triggers a Smart Contract to return the stored value to the search program.

## Evaluation Results

The goal of the evaluation is to test the functionalities and the performance of the implemented system. This includes the Document Register function and Document Query function. The evaluation will show the advantages of this concept compare to the system that rely on traditional database.

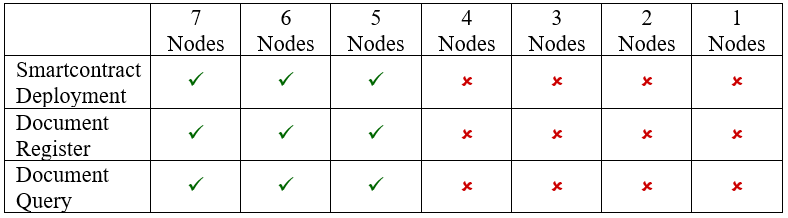


Table 1 Functionalities experiment result

Table 5 Functionalities experiment result

Table 6 Functionalities experiment result

Table 7 Functionalities experiment result

 Table 1 Functionalities experiment result shows that the system result from the implementation can function normally with 7, 6, and 5 active nodes. The system stops functioning when there are active nodes lesser than 5. After looking into the root cause of the result, it is turn out that the 7-Nodes Example cannot resolve the Block validation process cycle in a situation with fewer than 5 active nodes. When investigate the log, it shows that the Block validation process cycle which would repeat continuously and endlessly simply stopped at the first Block and there is no further upcoming process appeared in the log or none of any error notification appeared. Noteworthy, this is not even related to the condition of IBFT consensus where it requires at least two-third of all nodes to vote for the same Block version to complete the validation process. The system should be able to operate normally without error even there are fewer than five active nodes when all active nodes agreeing on the same Block version and there is no "bad actor node" present in the system to propose a falsified Block version that interrupts the vote. We think that a possible reason could be that the 7-Nodes example may not be developed for the situation with 4,3,2, or 1 active node as it was only built to aid Smart Contract developers to easily deploy and test their Smart Contract in the proper 7 active nodes simulated environment closely similar to the actual Blockchain network.

Chart, line chart

Description automatically generated

Figure 2 Processing time to complete Document Query   
varied by setup

The implemented system should be able to sustain a huge amount of data that continuously flows through the system without a failure. Each setup will be tested with the mockup transactions and measure the processing time required for the system to complete the specific process. There are ten transactions created for the experiment as test samples. In each transaction have its metadata attributes modified varied in each transaction for the test as. Each transaction will have about the same file size but different metadata attributes values to prove functionalities of the Document Query function. All experiments will be tested with these transaction samples resulting as 10 times test for each setup ("transaction number #n" will be referred to as "Document#n"). The evaluation will indicate the compatibility of the implemented system to an actual healthcare operation environment. <Explain Graph or should try performing more experiment?>

# Discussion

Amongst many metadata attributes, there are attributes that value can expose patients' confidential information into the Blockchain network. The attribute "sourcePatientInfo", for example, directly contains personal information for patients whose medical record the document is associated with. This attribute can contain multiple values such as patient name and address. So, in the actual adoption of the proposed concept, these metadata attributes values must be anonymized before entering the Blockchain ledger. This can be done by replacing the value with its hash counterpart when the attributes entering the Blockchain ledger via the Document Register function. At the same time, when the attributes were required for the search operation as search keywords input, the XDS Document Consumer will only need to hash the input value and use the hash to allow the XDS Document Registry actor to search for the matching hash value registered within the Document Registry Blockchain. This allows the concept to maintain confidentiality of patients' data while preserving the functionalities of the concept.

# Conclusion & Future Works

We achieved a system for the XDS Blockchain which acts as the medium for health document sharing amongst the XDS Affinity Domain Network. The prototype was designed to be compatible with normal XDS Actors while also act as a medium for a common XDS network to interface with the IBFT Blockchain ledger. This enhances the IHE XDS.b Profile with the Blockchain characteristics while appreciating the network to further share their health document to further benefit from the network for both operational interoperability and cyber-security.

Furthermore, the Smart Contract also has the potential to become the exchange medium for ITI-43 transactions where the XDS Document Consumer negotiates with XDS Document Repository for retrieving actual health documents, allow health documents exchanging activities in the network to be recorded in the Blockchain ledger which could be further used in the incident investigation during the cyber-incident. These would maximize the potential of Blockchain technology implemented on the Cross-Enterprise Document Sharing Profile.

# References

[1] Carestream Health, “Interoperability : Connecting the Healthcare Enterprise to Deliver Responsive Patient Care,” pp. 1–9, 2015, [Online]. Available: http://www.carestream.com/clinical-collaboration/sites/default/files/WhitePaper\_CCP\_Interoperability\_LTR\_201508\_en\_Web.pdf

[2] PolicyMedical, “Interoperability in Healthcare: To Have or Not to Have.” https://www.policymedical.com/interoperability-healthcare/ (accessed Sep. 22, 2018).

[3] A. Medical Association, “What is Information Blocking? Part 1”.

[4] IHE International Inc, “About IHE.” https://www.ihe.net/about\_ihe/ (accessed Sep. 11, 2018).

[5] IHE International Inc, “IHE IT Infrastructure ( ITI ) Technical Framework Volume 1 Integration Profiles,” *Int. J. Healthc. Technol. Manag.*, vol. 1, no. 8.0, pp. 1–177, 2008, doi: 10.1504/IJHTM.2008.017371.

[6] “First known ransomware attack in 1989 also targeted healthcare.” https://www.beckershospitalreview.com/healthcare-information-technology/first-known-ransomware-attack-in-1989-also-targeted-healthcare.html (accessed Oct. 29, 2021).

[7] “The 10 Biggest Healthcare Data Breaches of 2019, So Far.” https://healthitsecurity.com/news/the-10-biggest-healthcare-data-breaches-of-2019-so-far (accessed Mar. 04, 2021).

[8] “UPDATE: The 10 Biggest Healthcare Data Breaches of 2020.” https://healthitsecurity.com/news/the-10-biggest-healthcare-data-breaches-of-2020 (accessed Mar. 04, 2021).

[9] HIPAA Journal, “Largest Healthcare Data Breaches of 2018.” https://www.hipaajournal.com/largest-healthcare-data-breaches-of-2018/ (accessed Apr. 27, 2019).

[10] Healthcare IT News, “The biggest healthcare data breaches of 2018 (so far).” https://www.healthcareitnews.com/projects/biggest-healthcare-data-breaches-2018-so-far (accessed Apr. 27, 2019).

[11] Healthcare IT News, “The biggest healthcare breaches of 2017.” https://www.healthcareitnews.com/slideshow/biggest-healthcare-breaches-2017-so-far?page=1 (accessed Sep. 11, 2018).

[12] “Largest Healthcare Data Breaches of 2021.” https://www.hipaajournal.com/largest-healthcare-data-breaches-of-2021/ (accessed May 11, 2022).

[13] “The top data breaches of 2021 | Security Magazine.” https://www.securitymagazine.com/articles/96667-the-top-data-breaches-of-2021 (accessed May 11, 2022).

[14] Deloitte, “Key Characteristics of the Blockchain”, Accessed: Oct. 29, 2018. [Online]. Available: https://www2.deloitte.com/content/dam/Deloitte/in/Documents/industries/in-convergence-blockchain-key-characteristics-noexp.pdf

[15] D. Cosset, “The 4 characteristics of a blockchain - DEV Community.” https://dev.to/damcosset/the-4-characteristics-of-a-blockchain-2c55 (accessed Oct. 29, 2018).

[16] K. Peterson, R. Deeduvanu, P. Kanjamala, and K. Boles, “A Blockchain-Based Approach to Health Information Exchange Networks,” *Mayo Clin.*, no. 1, p. 10, 2016, doi: 10.1016/j.procs.2015.08.363.

[17] A. Ekblaw and A. Azaria, “MedRec: Medical Data Management on the Blockchain”, Accessed: Sep. 26, 2018. [Online]. Available: https://viral.media.mit.edu/pub/medrec

[18] G. Zyskind, O. Nathan, and A. S. Pentland, “Decentralizing privacy: Using Blockchain to Protect Personal Data,” *Proc. - 2015 IEEE Secur. Priv. Work. SPW 2015*, pp. 180–184, 2015, doi: 10.1109/SPW.2015.27.

[19] S. Tanwar, K. Parekh, and R. Evans, “Blockchain-based electronic healthcare record system for healthcare 4.0 applications,” *J. Inf. Secur. Appl.*, vol. 50, p. 102407, Feb. 2020, doi: 10.1016/j.jisa.2019.102407.

[20] D. Yaga, P. Mell, N. Roby, and K. Scarfone, “Blockchain Technology Overview (NISTIR-8202),” *Draft NISTIR*, p. 59, 2018, doi: 10.6028/NIST.IR.8202.

[21] “Home | ethereum.org.” https://ethereum.org/en/ (accessed Nov. 12, 2021).

[22] @wackerow, “Introduction to dapps | ethereum.org,” 2022. https://ethereum.org/en/developers/docs/dapps/ (accessed Dec. 04, 2022).

[23] V. Buterin, “A NEXT GENERATION SMART CONTRACT & DECENTRALIZED APPLICATION PLATFORM.”

[24] ConsenSys Team, “Quorum for Developers | ConsenSys.” https://consensys.net/quorum/developers/ (accessed Dec. 04, 2022).

[25] “PBFT and IBFT consensus - SettleMint Launchpad.” https://launchpad.settlemint.com/documentation/pbft-and-ibft-consensus (accessed Jan. 22, 2022).

[26] Quorum, “GoQuorum,” 2020. https://github.com/ConsenSys/quorum (accessed Jun. 03, 2021).

[27] “Node.js.” https://nodejs.org/en/ (accessed Jun. 10, 2021).

[28] “xml2js - npm.” https://www.npmjs.com/package/xml2js (accessed Jun. 10, 2021).