FAST DATA: A Fair, Secure and Trusted Decentralized IIoT Data Marketplace enabled by Blockchain

Akanksha Dixit*, Arjun Singh†, Yogachandran Rahulamathavan‡ and Muttukrishnan Rajarajan*
*School of Mathematics and Computer Science Engineering, City, University of London, UK
Email: {akanksha.dixit, r.muttukrishnan}@city.ac.uk
†School of Computer Science and Engineering, Nanyang Technological University, Singapore
‡Institute for Digital Technologies, Loughborough University London, UK

***Abstract*—As the world calls it, data is the new oil. With vast installments of Industrial Internet-of-Things (IIoT) infrastructure, data is produced at a rate like never before. Similarly, artificial intelligence (AI) and machine learning (ML) solutions are getting integrated to numerous services, making them "smarter". However, the data remains fragmented in individual organizational silos inhibiting data value extraction to it's full potential. Digital marketplaces are emerging to allow data owners to monetize this data. Yet concerns like privacy, security and unfair payment settlement deter adoption of such platforms. In addition, the state-of-the-art platforms are under the control of large multinational corporations with no transparency between buyer and seller in terms of payment details, listing, data discovery and storage. In this work, a novel decentralized platform of digital data marketplace for IoT data has been proposed. The platform leverages a decentralized data streaming network to host IoT data in a reliable and fault tolerant manner. The platform ensures fair trading, data storage and delivery in a privacy preserving manner and trust metric calculation for actors in the network. In order to study the feasibility of the proposed platform, an open source library is developed using Hyperledger Fabric and data network layer built on VerneMQ, the library is deployed on a real-time Google cloud platform. The library is tested and results are analysed for throughput, overheads and scalability.**

***Keywords*—** Industrial Internet of Things (IIoT), Data Marketplace, Blockchain, Smart Contracts.

I. Introduction

Sensing-as-a-Service has emerged as a rapidly growing industry spurring interest of enterprises that invested heavily in installation of Industrial Internet-of-Things (IIoT) infrastructure for past few decades. They have realized that data generated in their siloed environment is a tradeable commodity with resale value. With proliferation of AI based solutions in the market, data has garnered commercial value to train novel machine learning models. Federated machine learning models require IIoT data frequently in large volume. On the other hand, there are companies that are not inclined to invest in setting up IoT infrastructure but are willing to pay for reliable data streams from heterogeneous domains like weather, population density, soil type etc. They find it profitable to buy data from corporations that have such sensors installed. Such scenarios demand for a platform which makes it feasible to sell and purchase data seamlessly. However, wide-scale adoption of this idea still remains a challenge due to uncertainties around trust, cross-domain trading facility, fair transactions and security. Vendors willing to exchange data as a commodity are looking for assurance on validity of IoT devices and enterprises engaged in business. Siloed identity verticals, unauthorized malicious entity posing as data producers, centralized control of platform and fear around equitable payment settlement are challenges that require a new approach to make IoT data trading a reality.

In this work, a model is presented that is based on the *principle of data sovereignty* in which independent IoT business ventures can sell and purchase data reliably and in a trust-worthy manner with undisputed compensation to all the parties. We argue that such an effort requires a platform that disintermediates the central storage operator. In order to achieve transparency, accountability and fairness it is necessary to forgo centralized control from trusted third party (TTP) in terms of fair listing and seller discovery, identity validation and payment settlement. Therefore, the proposed digital marketplace uses blockchain platform as a decentralized trusted party that facilitates fair listing of data streams, identity verification and payment settlement using smart contracts. According to GDPR Articles 16 & 17 (Right to Rectification and Erasure), data modification/erasure must be possible to comply with legal requirements. Hence, critical data should not be stored on an immutable storage such as blockchain, which does not allow for modification of data, to ensure data integrity and trust. Therefore, in the proposed framework no private data is stored on blockchain. Moreover, blockchain is not a viable option for bulk data storage. Hence, the proposed platform leverages a decentralized data storage layer which is replicated across participating nodes to provide data delivery in a fault tolerant manner. The authentication of actors is executed using Self-Sovereign Identity (SSI) concept, wherein each entity holds a unique Decentralized Identifier (DID) [10] to authenticate on the platform.

Several blockchain based solutions are present in the market today for various industry verticals such as finance [9], healthcare [21], energy [12], automotive [33] etc. While data sharing marketplace have been proposed by startups [6], [34],

Design Elements	[1]	[2]	[22]	[23]	[27]	[36]	FAST
Smart-Contract	+	+	+	+	+	+	+
Decentralized Storage	-	-	-	+	+	-	+
Real-Time Data	+	+	+	+	-	-	+
Data Encryption	-	-	-	-	-	-	+
Trust Metric	-	-	-	-	+	-	+
Fair Settlement	-	-	part	-	+	+	+
Fault Tolerance	-	-	-	-	-	-	+
Decentralized ID Verify	-	-	-	+	-	-	+
DApp Support	-	-	-	+	+	-	+

Table I: Comparison of Blockchain Marketplace Solution

significant contributions have been made in the trust and fairness guarantees in the proposed platform. It is estimated that the value of IoT data marketplace will reach 3.6 trillion USD by the year 2030 [6]. Eliminating data sharing friction will serve industries like healthcare, smart city and advanced mobility infrastructure, agriculture, smart-grid and electricity, research etc and promote interoperability.

While solutions are available for data marketplaces focusing on centralized architectures using cloud infrastructure [14], this work attempts at designing and analyzing decentralized solution for IoT data marketplaces. The key contributions of this work are as follows:

- We propose a novel proof-of-concept decentralized solution to IoT data marketplace that supports actor verification, trust metric calculation and a secure and transparent transactional model for paying the sellers all in a decentralized manner by leveraging an innovative combination of blockchain, DID, peer-to-peer data stream storage and end-to-end encryption.
- The proposed model guarantees (i) fair settlement of compensation to all the involved parties, (ii) unbiased content listing using calculated trust metric, (iii) security of the content from peers hosting and delivering data streams (iv) fault tolerance and assured data delivery in case of faulty storage network peers.
- We develop a proof-of-concept implementation with distributed applications (DApp) and blockchain layer using Hyperledger Fabric, and the decentralized real-time stream data storage layer using VerneMQ [4]. The work includes study and discussion of the performance of the system which demonstrate that the system scales well with rational overheads.

The rest of this paper is organized as follows: Section II describes the related works. Section III gives defines problem definition. Section IV discuses the proposed architecture in detail with protocol flow and design analysis of the system. Section V describes the experimental settings and evaluates the performance. Section VI analyses the security of the proposed system. Section VII concludes the paper with discussion on future work.

II. Related Work and State-of-the-Art

Digital Marketplaces: A range of tangible and non-tangible goods are part of digital marketplaces today including commodity, e-books, music, images and videos. These marketplaces termed as eCommerce are not bound by physical boundaries and provide ease and competitive rates to consumers [20]. Such platforms have encouraged trading ecosystems like Software-as-a-Service (SaaS) and Infrastructure-as-a-Service (IaaS) [3]. However, since most of these digital platforms lack in listing and pricing transparency, seller has minimum to no right in determining the terms of licensing agreement. Moreover, these platforms are owned and regulated by a centralized authority which dominates the decision making process [19].

Peer-to-peer (p2p) data sharing network: P2P exchange of information has emerged as a growing fraction of world's digital economy [8]. This form of value exchange has expanded dramatically to include services, information trading and P2P lending. However, such networks suffer from regulatory challenges and building a marketplace using these platforms alone is challenging. Platforms such as BitTorrent [7] scale well yet building a marketplace on them is infeasible due to lack of properties like fairness, trust guarantees and payment settlements among network users.

Decentralized and Distributed Storage: As opposed to the centralized cloud servers, owned and regulated by a single organization, decentralized storage solutions consist of a P2P network, hosting and sharing data in a resilient and fault tolerant manner. Such form of storage has garnered attention both from research community and commercial organizations due to the advantages such as fault tolerance, collusion resistance and attack resistance. Decentralized storage solutions like IPFS [5] has inspired projects like Filecoin [17] and OrbitDB [25]. IPFS is a content addressable storage which prevents duplication and provides high availability of content. Storage platforms like IPFS when used along with a regulatory technology like blockchain can help in creating holistic solutions like Storj [31], Filecoin [17], Sia [30]. Although, these solutions have incentivization mechanism for participants, they lack in business logic, trust and fairness guarantees that are pivotal for a digital marketplace.

Decentralized IoT Marketplaces: The present state-of-the-art marketplaces are based on centralized cloud infrastructure owned by a single organization [13]. Due to research interest around decentralized marketplaces, there have been attempts to design marketplaces for data sharing [24], [18], [26]. However, marketplaces for IoT data remains a relatively untapped ecosystem. Xu et. al [36] explore the use case of blockchain marketplace in electricity and smart grid domain. They present a theoretical business model archetype for energy market in accordance with EU's Smart Grid Architecture Model (SGAM). However, their work lacks in a concrete architecture design involving incentive mechanism and security features. Missier et. al [22] propose a decentralized infrastructure for trading IoT data between producers and consumers exchanging a data unit "cube" with the help of trusted edge gateways. They analyse the feasibility using cost for "cube" settlement. However, it is difficult to establish the scalability and trust metric of the system. Ramchandaran et. al [27] present a

Figure 1: System Actors in the Proposed Framework

simple smart contract based implementation of a decentralized registry for IoT data trading for smart cities. However, their work largely overlooks the analysis and illustration of essential elements of a data marketplace like payment scheme, trust and privacy. Niavis et. al [23] proposes a decentralized data sharing infrastructure for off-grid networking using blockchain network and a distributed file system. While their work takes into consideration a great deal of factors like identity management and private data exchange, elements like payment settlement, trust among trading entities and fairness remain unexplored. Drawing motivation and understanding from the limitations of discussed works, an attempt is made to design a fair, transparent, reliable and trust-less system for IoT data trading in a decentralized manner.

III. System Actors and Problem Definition

A. System Actors

The actors as shown in Figure 1 with their respective roles in the system as as follows:

1) **Seller** (S) wants to use the marketplace to trade the IoT data generated in it's facility. It may not be able to host the data itself and create an infrastructure around it. Hence, it uses the service of digital marketplace to monetize it's otherwise siloed machine data. S wants the buyers to discover it's offers and pay fairly for the data. S is willing to pay the storage operator nodes in exchange for their data hosting and processing service.
2) **Storage Operators** (O) are data facilitators of the system. They are a cluster of decentralized peer-to-peer (p2p) nodes that host data in the form of data streams and are incentivised for their service. The independent server providers with stake in the system can choose to act as storage operator nodes and provide storage service in the system at competitive rates.
3) **Client** (C) comes to the marketplace platform to browse, purchase and get access to a myriad range of on-demand data sets which otherwise is difficult for them to procure in a trusted manner.

B. Problem Definition

The problems addressed in this paper are to ensure fair, secure, trusted delivery of IoT data from a seller to a client in a decentralized manner. Let Π be a protocol executed by a cluster of storage operator nodes $\mathbb{O}$, with each node $O_i \in \mathbb{O}$ hosting data for multiple topics $\mathbb{T}$ and a client downloading a batch d of data stream of a topic $T_i \in \mathbb{T}$ and paying price p.
Fair Trade: Protocol Π is fair if the following hold:

1) **Client Fairness:** If C pays p according to Π, it is assured of receiving the subscribed batch d of data stream of topic T_i for the agreed mode and time period before termination of Π.
2) **Seller Fairness:** If C pays p according to Π to purchase a batch d of data stream of topic T_i published by P, then P is assured of getting their share of payments.
3) **Storage operator Fairness:** For all $i \in |\mathbb{S}|$, if S_i gives access to d of T_i according to Π, it is assured of receiving payment.

Privacy: Protocol Π is considered secure if execution of Π does not disclose the data on the marketplace to any storage operator. The access of data should be given only to a client who has purchased it by making payment p. It is important to note that without privacy guarantee, fairness cannot be assured, as any storage operator that gains access to data without prior payment is a violation of the Π.
Trusted Exchange: Protocol Π is considered secure if it allows only verified sellers to host data on the platform. After verification, an honest seller should be rewarded with higher Trust Score ($\mathbb{TS}$) based on transaction history and feedback from trading parties.

IV. Proposed Platform

A. Design Features

In the proposed model, the storage operator nodes are synchronized with each other to serve data under various topics. Each topic is replicated to multiple partitions to avoid centralization and ensure fault tolerance and scalability. Given the above actors in the system, several questions arises with regards to how the funds flow in the system, the payment scheme and intervals of payment. While there are many variants possible, two broad variants are discussed below:

- **Batch Mode:** In this mode of trading, the client can access IoT data in batches at regular intervals for a stipulated period of time. The agreement in this mode involves payment of a flat fee periodically before delivery of each batch of data. We call this "batch" mode as data is not served in real-time rather as small packets of information.
- **Stream mode:** This mode is meant to serve data as real-time streams subscribed by the client. The client pays the seller and storage operator periodically for the subscribed streams. The payment is made in advance by the client before the streaming of purchased data can begin.

The proposed solution works with both the variants where a smart contract ensures fair and smooth data trading. It achieves a reliable trading flow in which the seller and storage operator is paid for data and service respectively whereas the client is assured of data delivery without losing any deposited payment. In case of failure to deliver data, the smart contract refunds the remaining funds back to the client.
Identity: Identity verification plays an important role in the framework as it contributes towards trust building and reputation management of the clients. An identity verification step is mandatory for seller in the proposed work. However, a client can opt to undergo DID verification. If a client chooses to

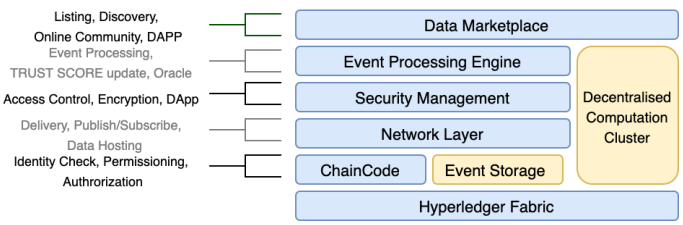

Figure 2: Layered Architecture of proposed Framework

undergo the verification, it will have a wider range of data streams to purchase as some sellers might want to give access to their data to verified clients only. It facilitates benefits for sellers (in the form of ability to charge high rates for verified data) and benefits for buyers (in the form of discounts and download permission for critical data).

We assume that the participating entities are adopters of decentralized identity, Self-Sovereign Identity the (SSI) framework [35]. In this framework, the end-users select a key-pair based on asymmetric cryptography and call it DID (did), which is associated with digital credentials issued by issuers such as government, hospitals, universities, corporations etc. This ensures that each seller on the platform is a verified organization/agency/facility. The process is very similar to an e-Know Your Customer (eKYC) process done by an organization to onboard customers.

Trust: The marketplace maintains a TRUST SCORE ($\mathbb{TS}$) metric for every actor in the network as explained later in this work. $\mathbb{TS}$ is derived from two primary factors, reputation and credibility. No centralized authority is involved in trust score calculation.

Data Security: In our work, we ensure that data originating from a seller is encrypted before it is outsourced to a storage operator and later decryption keys are only available to a client who has made upfront payment. We apply end-to-end encryption (E2EE) on IoT data in our design. All the communications are TLS/SSL encrypted.

B. Components of Layered Architecture

This section discuss the roles of each layer from Figure 2.

1) Data Marketplace: Data marketplace is a listing portal implemented as a DApp. DApp is a decentralized application/website that interacts with the blockchain back-end through smart contracts. It enables listing and discovery of data offers (DO) from the sellers. Data is offered in a publish-subscribe data model [29] as it is suited for IoT domain, since both the sellers and buyers are potentially large and are constantly changing. A DO will contain meta-data for each data stream like topic, the payment mode associated with data subscription mode, price and terms/conditions for a buyer to purchase the data stream (verified buyer) etc as shown below.

$$Data\_Offer = Topic + PaymentMode + Price + Terms$$

In the marketplace, listing will be available from potential sellers accompanied with their respective $\mathbb{TS}$.

2) Event Processing Engine: Events are generated from various sources for e.g. at the origin of data streams, from DApps, from client applications, execution of Smart Contract functions etc. Engine listens to the events on network, processes them for consumption by DApps/Smart Contracts and take real-time actions on them. It also processes events to update the $\mathbb{TS}$ for the entities.

3) Security Manager: This layer envelops the storage operator nodes and monitors all the communication made by an entity with them. The seller encrypts the data stream and upload it on storage operators through this DApp. Later, when a client wants to download the data, it first verifies its identity with the smart contract, makes payment and obtains an access token. This layer verifies the token with the blockchain and allows the client to access data stream for allotted time frame.

4) Network Layer: It is the primary data transport layer of the architecture. It is composed of a collection of p2p network of storage operator nodes that facilitate data streams and decentralized management of events. These nodes are compensated for their service. The layer uses policies and permissions defined in the smart contract for its functioning.

- *Data Streams:* Encrypted data streams are hosted by verified sellers on the storage operator nodes. A single stream is replicated across multiple storage operators to ensure fault tolerance. To host a data stream, seller makes a data offer and registers itself on the platform. When a client wants to purchase data, it enters into a agreement with the seller, the terms of which are detailed down in the form of a Data Sharing License. The fields of data stream are *streamID, streamName, topic/type, owner/seller, permissions, conditions*.
- *Storage operator Nodes:* They host data streams, deliver encrypted data to clients, forward event streams to the processing layer and communicate with security manager DApp to securely process authorization of clients requesting data stream access. Storage operator nodes will expose all these functionalities via APIs.

5) ChainCode/Smart Contracts: These are self-executing computer programs that enforce and govern the terms of agreement in a blockchain. In the proposed architecture, smart contracts are used for coordination between sellers and buyers, identity verification (using SSI), payments, incentivization and permissioning.

C. System Architecture Overview

A centralized data marketplace can encounter critical challenges like ensuring fair trade settlement, privacy and security guarantees [28]. A decentralized blockchain network of consensus abiding peer nodes can supplant the TTP, enforce equitable trading rules and manage autonomous payments along with transparency and non-repudiability. A data storage layer is required to host data and supplement blockchain layer in the marketplace. The storage layer lacks intelligence and need additional services to monitor security and privacy policies. The *Security Manager* DApp regulates the authorized access to stored data on this layer. It allows seller S and client

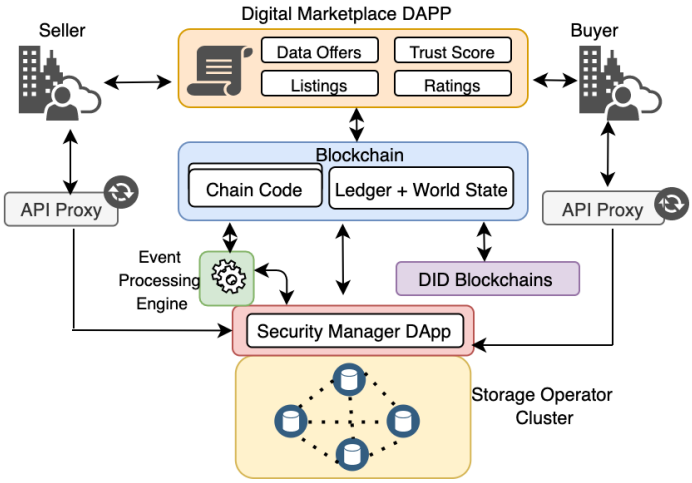

Figure 3: Detailed Architecture of Proposed Data Marketplace

C to interact with the storage layer only after authorization. Seller S and client C will communicate with the blockchain through Data Marketplace DApp to invoke ledger to make queries and later transactions. The use of blockchain in proposed architecture remains very generic and most systems with support for smart contract can be used. However, a permissioned blockchain with robust support for identity and access control is better suited for an enterprise application. In this work, we use Hyperledger Fabric as the blockchain platform. Figure 3 shows a detailed architectural diagram with all the system components.

D. Notations and Preliminaries

The blockchain ledger is denoted as $\mathbb{L}$. All the transacting parties will register with $\mathbb{L}$ before any trading takes place on the platform. *KeyGen* $(1^{\lambda}) \rightarrow (sk, pk)$ generates a private key sk and public key pk given a security parameter λ. Each actor will implement this algorithm to generate the key pair.

Data streams are identified by their unique *streamID* which includes *seller's name (S), stream name* and *time-stamp* of origination. Let GenerateID(.) be the procedure that returns a *streamID* given a batch d of data stream, $streamID_d \leftarrow$ GenerateID$(S||streamName||timestamp)$.

HMAC-SHA256(k,P) is a keyed-hash message authentication code where k is the secret cryptographic key and P is the plain text for which *MAC*, σ needs to be generated. $S_{sk}(m) \rightarrow \phi$ is a signature algorithm that generates the signature ϕ of the message m with the private key sk, and $V_{pk}(\phi, m) \rightarrow \{0,1\}$ is a verification algorithm that verifies whether the signature ϕ on message m is valid or not using the public key pk and returns 1 if ϕ is a valid signature or 0, otherwise. Let (Enc, Dec) be a secure symmetric encryption scheme. This scheme will be used by the sellers to encrypt the data stream 'd' before uploading encrypted data stream 'e' on storage operator nodes.

E. Smart Contract Functions

There are two DApps through which smart contract functions are invoked. These functions are categorized as follows:

1) Data Marketplace DApp
   - *RegisterAndVerifySeller*(did_S, pk_S): seller S sends its DID and public key.
   - *RegisterAndVerifyClient*(did_C, pk_C): client C sends its DID and public key.
   - *RequestUploadToken*(DO, pk_S): S requests authentication token to upload DO on storage operator nodes.
   - *RecordStreamDetails*$(streamID, DO, k_m, k_e, \sigma_d)$: record stream details on blockchain.
   - *RequestDownloadToken*(p, DO, pk_C): C makes payment p and requests exchange token.
2) Security Manager DApp
   - *UploadDataStream*(ϕ_S, pk_S, e): S send encrypted data e with signed authentication token.
   - *DownloadDataStream*(ϕ_C, pk_C): C sends signed exchange token to receive data from storage operator nodes.

F. Protocol Flow

The protocol is divided into three phases. The onboarding phase deals with verifying the identities of the actors in the network using did and registering them in the marketplace. The data upload phase will allow the sellers to complete the necessary steps to get their data up in the market for sale. The data purchase phase will explain the steps involved from the time a client picks a data offer to the settlement of payments.

1. Onboarding Phase: In this phase, the entity seller/buyer makes an account, verifies its identity and registers itself for trading activities on the blockchain through marketplace DApp. Each entity uses *KeyGen* $(1^{\lambda}) \rightarrow (sk, pk)$ to obtain a key-pair which is used to obtain a DID (did). The actor sends the did associated with its identity to the DApp, which is forwarded to the blockchain. The blockchain resolves the did through the underlying decentralized network/blockchain which was initially used to issue it to the entity. If the did is successfully verified, an account is created for the actor and it's $\mathbb{TS}$ is incremented on the DApp. Later, the seller registers the data offers DO it wants to make in the marketplace. These offers along with payment details and terms are recorded on the blockchain.

Algorithm 1: Onboarding/Actor

```
func VerifyandRegister(did, pk){
Send did, pk to Data marketplace DApp
If ID verified with issuing blockchain, register on 𝕃
S uploads DO on Data Marketplace DApp }
```

2. Data Upload Phase: The seller S generates $streamID_d$ which uniquely identifies d. Then, S generates a key K_m for obtaining message authentication code of the data, *MAC(d,* k_m*)* $\rightarrow \sigma_d$, so that any potential client can verify the integrity of purchased data. Later, S generates a secret key K_e and encrypts data batch *Enc(d,* k_e*)* $\rightarrow e$. Once the data is ready, it initiates data upload by sending a request to marketplace

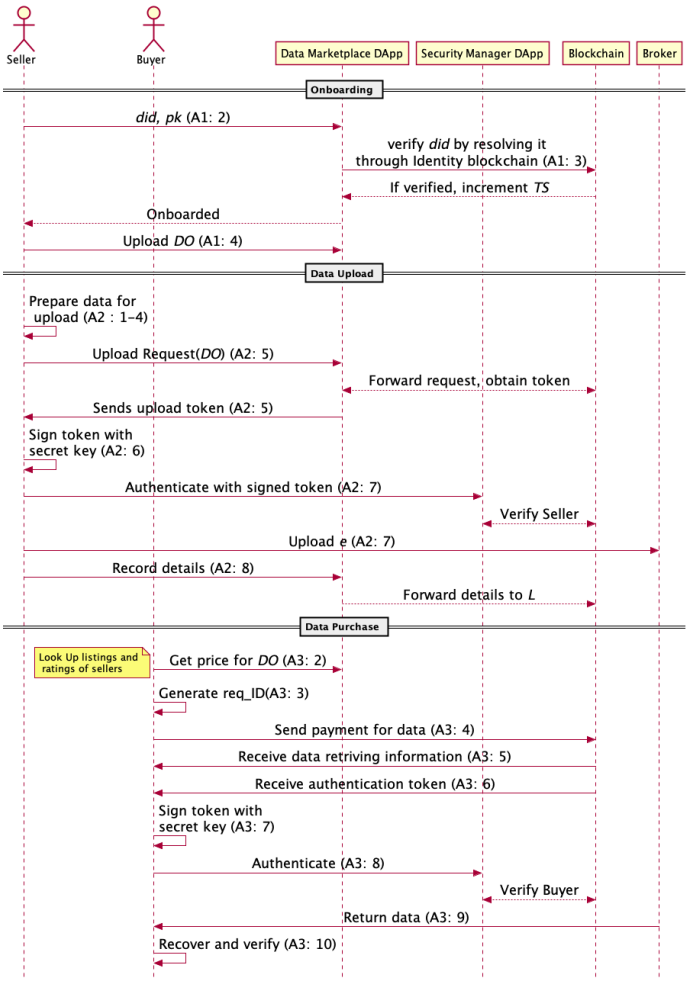

Figure 4: Protocol Flow

Algorithm 2: Data Upload/Seller (S)

func **DataUpload**$(d, DO, k_m, k_e, S, streamName(sN), timestamp(ts))$ {
streamID$_d \leftarrow$ *GenerateID*$(S||sN||ts)$
MAC$(d, k_m) \rightarrow \sigma_d$
Enc$(d, k_e) \rightarrow e$
Upload Request (DO) $\leftrightarrow \omega$
$S_{sk}(\omega) \rightarrow \phi_s$
Verify token $V_{pk}(\phi_s, \omega) \rightarrow$ {0,1}, upload data e to B
DataAdded$(streamID_d, DO, k_m, k_e, \sigma_d)$
}

DApp. The DApp sends the request to the blockchain which responds by sending a token ω to S. The token is used to uniquely identify an upload request by the S, therefore, it signs ω with its secret key sk to generate ϕ_S. Using this signed token, S verifies with the *Security Manager* DApp through *API Proxy* layer, a service to help client interact securely with storage layer. The encrypted data e along with related metadata is stored on B, and secret keys, σ_d, *streamID*$_d$ and corresponding DO are recorded on the ledger.

Algorithm 3: Data Purchase/Client (C)

func **DataPurchase**$(DO, streamID_d)$ {
Get p for $streamID_d$ from $\mathbb{L}$
Generate a request ID, $reqID$
Send **Payment**$(streamID_d, reqID, p)$
Get K_m, K_e, σ_d from $\mathbb{L}$ for $streamID_d$.
Receive δ from $\mathbb{L}$ to retrieve data from B
C signs token $S_{sk}(\delta) \rightarrow \phi_c$
Authenticate at DApp $V_{pk}(\phi_c, \omega) \rightarrow$ {0,1}
If verified, receive e from B
$d' \leftarrow Dec(e, k_e)$, $\sigma_d' \leftarrow MAC(d', k_m)$. If $\sigma_d' == \sigma_d$, $d' == d$, data verified
}

3. Data Purchase Phase: In this phase, C browses through the listing of available DO on the DApp. It selects a batch d of data with unique *streamID*$_d$ on the marketplace. It generates a request identifier *reqID* and sends it to the blockchain with the payment p for the requested data stream through marketplace DApp. The ledger will first verify if the owner S has placed any restriction on the client it wants to sell data to, for e.g. the client C needs to be a verified client. If the terms specified in the DO are met and payment p is received successfully, the process to establish data transfer is initiated. C receives two secret keys from the ledger, encrypted by its pk, one for verifying data integrity and other for decrypting the data. Later, blockchain sends an exchange token δ to C. The δ is signed by C using its sk to generate signature ϕ_C. C approaches the *Security Manager* DApp through *API Proxy* service with ϕ_C. Upon successful authentication, it receives the data. It decrypts the data, and creates a MAC from recovered data, if the generated MAC matches with received one, the client can be assured of integrity of data.

Payment: In section III-B, we have defined the term fairness from a seller, client and storage operator perspective. In the proposed design, the client can fetch the requested data stream only after making payment to the blockchain. Since, the interaction between the client and a storage operator contains an intermediary layer, *Security Manager* DApp, ascertaining that a storage operator has actually served the data it easy. This layer keeps a check on the malicious storage operators who otherwise can claim to have served the data without doing so in order to maximize profit. Similarly, in our design it is difficult for a malicious client to claim that they have not received the data, while they have already downloaded it.

G. Design Analysis

In this section, we demonstrate that the proposed design satisfies the goals outlined in section III-B. It is to be noted that since a storage operator O is randomly assigned to S, a collusion among the client C and storage operator O_i is unlikely. Also, there is no direct interaction between S and C in the proposed protocol, hence their collusion cases

reduce to individual malicious behavior rather than a group phenomenon.

Fair Trade

Client Fairness: The proposed protocol Π ensures that an honest client who has made payment p to $\mathbb{L}$ should receive access to data stream d for agreed time frame. Specifically, it should be able to access the data stream d and it's respective decryption and integrity-check keys. Under the assumption that blockchain is tamper proof and storage operator nodes do not have direct access to unencrypted data, honest storage operator nodes will follow the protocol and keep serving the data stream. Since, the execution of smart contract is autonomous, tamper-resistant and unbiased, therefore, malicious parties cannot effect the key release by $\mathbb{L}$ after a successful payment made by C. Thus, Π guarantees client fairness.

Seller Fairness: For every successful data exchange, the respective payment is made to $\mathbb{L}$. As the smart contract holds logic to directly pay S, execution of Π guarantees seller fairness under the assumption that smart contract execution cannot be tampered with.

Storage Operator Fairness: In the proposed protocol, each storage operator O_i when delivers a data stream d for agreed time frame, an event is generated and it receives the payment from $\mathbb{L}$ corresponding to it. Under the assumption that the smart contract execution is tamper-resistant, the payment to all storage operator nodes is guaranteed as it is controlled by $\mathbb{L}$, only when they served C. Thus, the protocol Π guarantees storage operator fairness.

Privacy: According to Π, no storage operator can look up the data stored on storage operator p2p network. This is due to two reasons, first, due to E2E encryption, nodes are oblivious to the content of the data, and secondly because keys are not available to any storage operator node. Therefore, a disconnect is maintained between the data and key set, which can only be simultaneously obtained by a client C who has made upfront payment to the smart contract. Under the assumption that smart contract is fairly deployed, an adversary will not be able to obtain the data. In this way, privacy and security of the data is maintained by Π.

Trusted Exchange: When multiple unknown parties transact with each other, there should be a metric to ascertain their honesty and integrity. In the proposed system, we use TRUST SCORE ($\mathbb{TS}$), a metric associated with each seller and client on the platform to assess their trustworthiness. The trust metric of PeerTrust's in [16] is modified to suit the proposed platform's functionality. We use reputation and credibility values to compute the trust values. In this work, reputation is derived from the verification status of entity's identity which we denote by decentralized identifiers (DIDs). Credibility represents the confidence an entity has garnered from its transacting parties. It's value is derived from ratings and feedbacks an entity received after completion of each transaction in the past.

Given a recent time window, let $I(u, v)$ denote the total number of transactions performed by an actor u with v and I(u) denote the total number of transactions performed by actor u with all other actors, $a(u, i)$ denote the other participating actor in actor $u's$ ith transaction, $S(u, i)$ denote the normalized amount of satisfaction actor u receives from $a(u, i)$ in its ith transaction, $Cr(v)$ denotes the credibility of the feedback submitted by v, $TR(u, i)$ denote the transaction rating v gives to u for ith transaction. For an actor v, actor u's credibility at a given time can be computed using following equation:

$$Cr(u) = \sum_{i=1}^{I(u)} (S(u,i)) * Cr(a(u,i)) + TR(u,i) \quad (1)$$

The actor u's $\mathbb{TS}$ at a given time can be calculated as:

$$TS(u) = \sum_{i=1}^{V(n)} VI(k,u) * Cr(u) \quad (2)$$

where $V(n)$ is the total number of identifiers submitted by actor u, $VI(k, u)$ is the reputation value derived from the identifier, organization k issues to u, and $Cr(u)$ is actor u's credibility as calculated in equation 1.

External Sharing: In cases where seller has no objection to reselling of the data, they will define terms of reselling in Data Stream Licenses as defines in Section IV-B-4. However, when Seller does not want to resell the data, a game-theory based pricing model nees to be developed to make external data sharing economically unattractive. Such a practice will be discouraged as in order to make profit, the party will quote a higher price for reselling data. Whereas, the marketplace always offers a wide variety of data at competitive prices. Yet another inhibiting factor is that IoT data, unlike other formats of data (videos, software, images) derives its value from real-time generation and consumption. Therefore, for a malicious party it will be increasingly difficult to get hold of a constant stream of IoT data at profitable prices from a party externally sharing data without delay.

V. Implementation and Result Evaluation

A. System Components

1) Hyperledger Fabric: The proposed framework is blockchain platform agnostic as long as it supports execution of smart contracts and some form of transaction processing [15]. However, for an enterprise use-case certain level of confidentiality is preferred as compared to a completely public system (Blockchain, Ethereum). In a public blockchain, anonymous participants join the network without validation and view the complete transaction details which makes the network fully untrusted. Another reason remains poor transaction throughput for public blockchains like Ethereum and Blockchain. Hyperledger Fabric (henceforth called Fabric), serves the requirement well. It is an open source, modular permissioned blockchain platform that permits pluggable components for consensus, data store and membership service management among other modules.

assume there are three organizations (a private firm, a government agency and a research lab) each contributing peer nodes as resource in the network. For the governance structure,

in Fabric we have a Membership Service Provider (MSP), that binds identity of each participant to a root identity provider. Each of these three organizations will also define an *admin peer* using their MSP which define policies and roles in the network. And the network is governed by these policies including transaction ordering, execution and processing.

Each Fabric peer in the network runs an instance of the Data Marketplace smart contract. This contract maintains the listing of data offers including terms and mode of data streams, client details and provide interface to make and verify payments. The Security Manager smart contract deals with authentication and access management to access the data streams from VerneMQ network. In the results section, for blockchain nodes = 3, each organization contributes one peer each and for nodes = 6, the contribution is two from each organization.

2) VerneMQ Cluster: VerneMQ [4] is a distributed MQTT(Message Queuing Telemetry Transport) message broker, that ensures low latency and fault tolerant guarantees. MQTT is a lightweight, publish-subscribe network protocol that transports messages between devices. The encrypted data streams are stored on the VerneMQ network nodes that are modelled in a crash tolerant fashion. As a whole, VerneMQ provides a reliable data hosting capable of tolerating faults. VerneMQ with some additional security access policies constitute the storage layer in the proposed architecture.

3) Interaction Details: Data marketplace and Security Manager DApp are implemented as Golang application interacting with blockchain network through smart contracts. Sellers and Buyers are implemented as Golang scripts that sends queries and transactions to blockchain network by requesting the respective application. Both these entities publish and subscribe to data streams by authenticating at security manager DApp.

B. Evaluation

The experiment was setup on Google Cloud Computing Platform (GCP) running an instance of Linux Ubuntu-2004-focal-v20210325 configured with 16vCPUs (16 CPU cores), 64 GB RAM and 100 GB HDD storage. The experiments were run by varying a number of parameters to understand the dependency on each of them. The parameters being number of storage operator nodes (i), number of blockchain nodes per organization and file size to be uploaded and downloaded. In each experiment, we vary a different parameter keeping other parameters constant, to understand its impact on performance. The Fabric network was run in the form of Docker containers. Each experiment is run 5 times and the averaged observations are reported. To analyse the performance of experimental setup, a benchmark tool was written in Go v1.15.0. The benchmark tool helps to create several virtual sellers and buyers that sends simultaneous requests to the blockchain network. We chose to implement the IoT devices communication using MQTT as its a lightweight protocol and closely resembles the communication requirement of an IoT data marketplace. The IoT data used for the testing was generated by a virtual clients written in Golang. Depending on the file size given as input

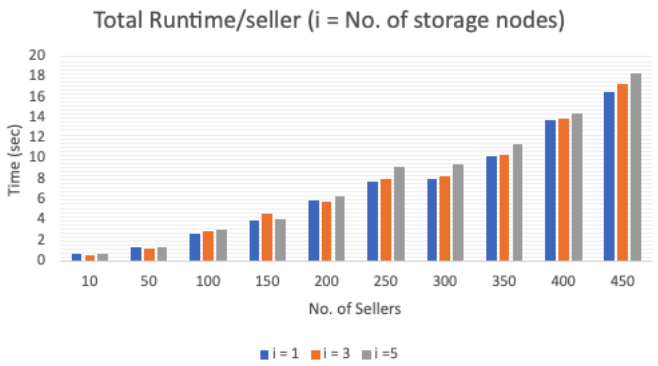

Figure 5: Total Runtime/Seller [Blockchain Nodes = 3]

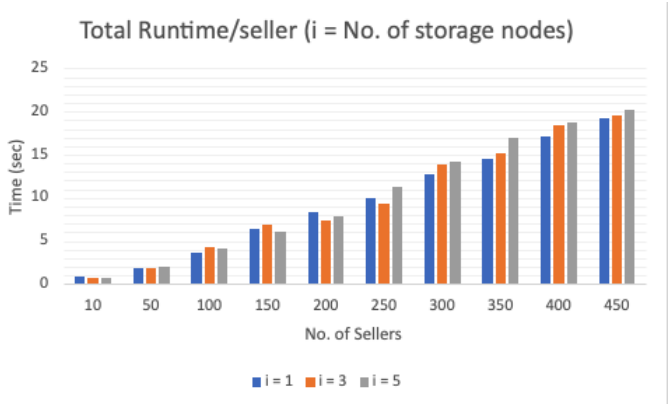

Figure 6: Total Runtime/Seller [Blockchain Nodes = 6]

by the user, a script recursively generated bytes of data at constant interval to emulate IoT sensor data generation from a hardware sensor. The complete open-source implementation of data marketplace project is available online at Github [11].

Varying concurrent load We study how latency observed by a Seller varies with overall system load both for blockchain node count 3 and 6 each for storage cluster (i) count as 1, 3 and 5 . We measure the latency as two parameters i.e. total runtime for each seller as shown in Figure 5, 6 and upload runtime as shown in Figure 7 and 8. The total runtime includes the time taken by a Seller to register on the blockchain network and adding data offers on the network while upload time is the time taken by a Seller to authenticate and upload data on storage operator network. It can be inferred from the results that with the increase in the number of concurrent sellers, the total runtime increases. However, total runtime is a one time process and therefore, it is a feasible overhead for each seller in the network. The upload runtime is relatively low when compared to the total runtime which is a good practical advantage as data upload will be done more often by the sellers.

Varying Blockchain Nodes It can be seen from Figures 5 and 6, that with increase in the number of blockchain nodes, total runtime do not come down, rather it increases by fraction of milli-seconds. This is due to the fact that with increase in the

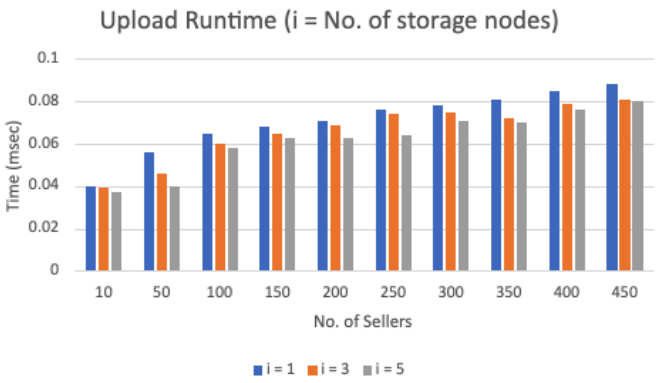

Figure 7: Average Upload Time [Blockchain Nodes = 3]

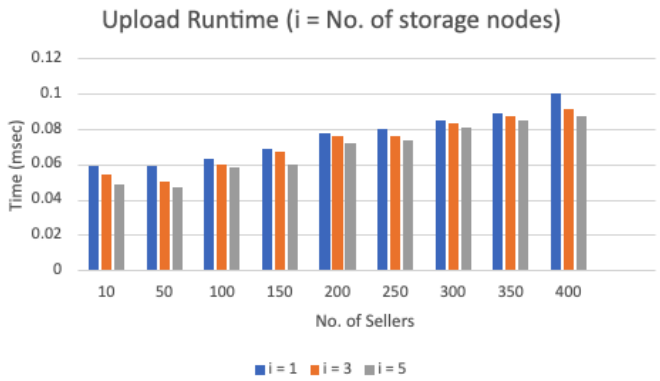

Figure 8: Average Upload Time [Blockchain Nodes = 6]

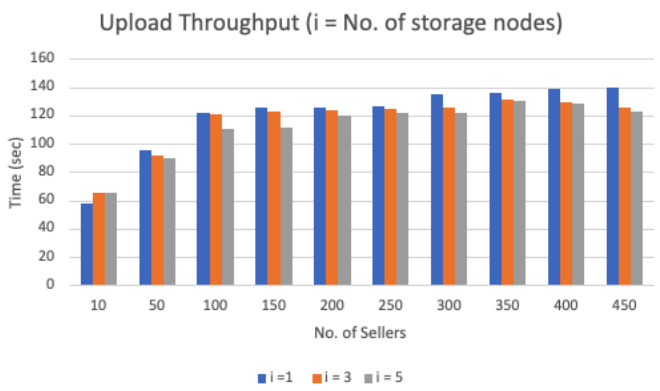

Figure 9: Upload Throughput [Blockchain Nodes = 3]

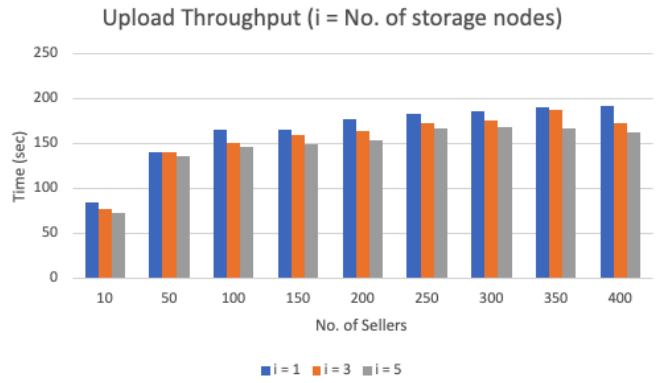

Figure 10: Upload Throughput [Blockchain Nodes = 6]

number of blockchain nodes, the transaction synchronization time among nodes increases. The real bottleneck is transaction processing, and not the consensus protocol as elaborately explained in the work [32]. The transaction synchronization time can be reduced by vertical scaling of the system i.e increasing the processing power. However, this only comes into effect for transactions that lead to state change in blockchain and not for queries to the blockchain as seen in Figure 7 and 8.

Varying number of storage operators (i) The results for variation in the number of storage operator cluster can be witnessed for all the results. The upload runtime shows positive trends for average upload time with increase in the value of *i* as shown in Figure 7 and 8, which will improve the publishing speed of the sellers hence increasing the overall throughput as shown in the Figure 9 and 10. Therefore horizontal scaling helps in the case of storage operator nodes. While for blockchain nodes, vertical scaling along with sparse peer node that selectively commits transactions as proposed in [32] will be advantageous. Upload throughput Figure 9 and 10 is the total bandwidth (msgs/sec) processed by the system under concurrent load of Sellers. The upload throughput shows similar performance trends as total runtime/seller in Figure 5 and 6. However, the performance increases with increase in number of storage operators.

Varying file upload frequency We study the impact of data stream sizes on latency faced by the clients. Figure 11 and 12 shows the latency impact for upload of data streams, for file sizes ranging from 100 B/sec to 1 MB/sec for blockchain nodes 3 and 6 both. As expected, the latency increases sub-linearly with file size in both the cases but the overall overhead shows positive outcome. The overhead of file size increase is within 10% of the baseline.

VI. Security Analysis

In this section, we analyse the most common security threats in IoT (MITM, DDoS) and blockchain domain (Sybil, Forking). In traditional IoT systems there are many security weaknesses, some of which can be removed with the use of blockchain such as information disclosure, repudiation, dependency on a trusted third party. The discussed attacks have been specifically chosen as they are critical from a decentralized architecture point of view such as IoT.

A. Man-in-the-Middle (MITM) Attack

In order to prevent MITM attack, several defense mechanisms have been provided in the architecture. Firstly, all the communications taking place between architecture components happen over a secure channel over TLS/SSL. Secondly, all the transactional messages are digitally signed with the secret key

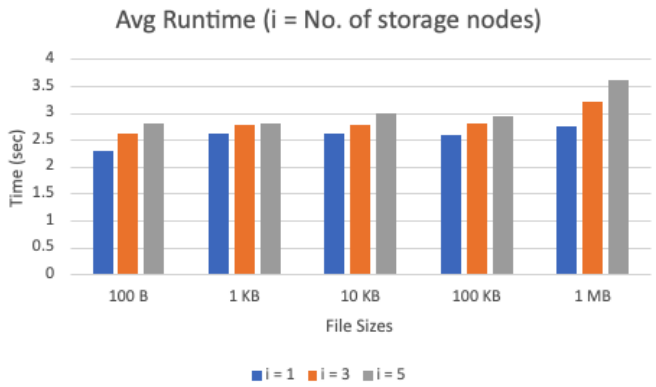

Figure 11: Average runtime with file size variations for [Blockchain Nodes = 3]

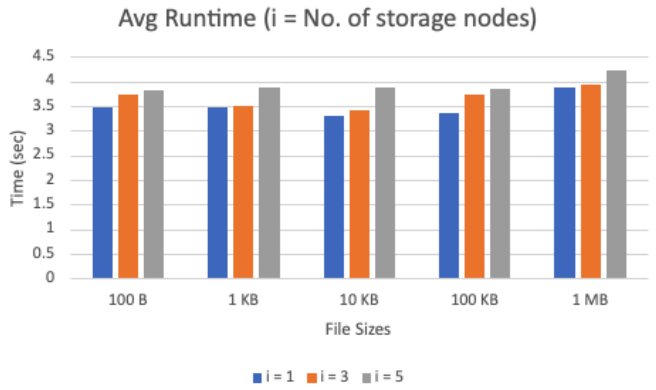

Figure 12: Average runtime with file size variations for [Blockchain Nodes = 6]

sk of the sender which makes it difficult for an impersonator to replicate them. Besides, since the sellers and clients in the network posses unique did, secured by public-private key pair, an attacker cannot inject malicious signed messages.

B. Distributed Denial of Service (DDoS) attack

DDoS attack can be mitigated by using unique identities in the network as proposed in our architecture. The blockchain network in general provides a certain level of security against malicious nodes. Moreover, since the blockchain in our architecture is a permissioned network hence, only trusted nodes are added in the network. In addition, only entities whose dids have been successfully verified are allowed make account and later query/send transactions in the network thus, preventing malicious traffic attacks on network.

C. Sybil Attack

Sybil attacks are a major challenge in public decentralized network where the same node can fake multiple identities. The proposed framework has an advantage over this attack as the blockchain network is a permissioned network. In this network the consensus over a set of transactions is achieved only after receiving endorsements by a number of verified nodes. Hence, no single node or a set of nodes holds absolute power. In the Fabric blockchain, identity certificates are issued to each node and hence, identity faking becomes difficult.

D. Information Disclosure Attack

In the proposed framework, the information stored on blockchain can strictly be queried only by verified actors of the system. For e.g. a client can only request key release once they have made full payment for the data. In order to query DO which includes price, metadata and terms of data exchange ,an entity needs to verify their did. Similarly, no suspicious entity can gain access to the data stored on storage operator nodes until they authenticate themselves at *Security Manager DApp* using authentication token.

E. Forking Attack

Hyperledger Fabric blockchain differs from well-known public blockchain like Bitcoin and Ethereum in the sense that in Fabric the consensus among nodes is a three step process which includes execute-order-validate unlike order-execute. First the transactions are checked for correctness and endorsed by endorsing peers. Then, they are ordered via a consensus protocol and lastly they are validated according to application-specific endorsement policy before committing transactions to the ledger. Hence, the proposed architecture can prevent forking attacks.

VII. Discussion and Future Work

In this work, we addressed the issue in real-time decentralized trading of IoT data which is a future reality owing to the massive progress in advanced AI algorithms that need bulk of data to make informed decisions. By leveraging an innovative combinations of blockchain, peer-to-peer storage facility, decentralized applications, the proposed system guarantees fairness to all participants despite presence of maliciousness, privacy of data from peers involved in hosting content, fault tolerance and availability, all without reliance on a central facilitator.

While designing such a system, certain challenges were faced that can be taken up as future work in this research direction. The first challenge remains designing equitable payment methodologies between the two parties. The parties can agree to transact in either cryptocurrencies/native token or existing payment methods. Crypto-tokens can also act as incentive mechanism for early adopters of the platform. Designing a payment layer over such a system architecture is one of the future works. Yet another challenge remains to put a check on external sharing of platform IoT data in the long run. For that, game theory approaches can be taken to design, better pricing models and terms of data exchange. In the proposed work, we have proposed a basic incentivization mechanism, in future works, more robust and theoretically tested schemes can be designed to make unfair transactions economically unattractive. These directions are some of the challenges of the proposed framework and they hold huge possibilities for improvement.

References

[1] K. R. Ãzyilmaz, M. DoÄan, and A. Yurdakul. "IDMoB: IoT Data Marketplace on Blockchain". In: *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*.

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

[3] P. Banerjee, R. Friedrich, C. Bash, P. Goldsack, B. Huberman, J. Manley, C. Patel, P. Ranganathan, and A. Veitch. "Everything as a Service: Powering the New Information Economy". In: *Computer* 44.3 (2011).

[4] M. Bender, E. Kirdan, M.O. Pahl, and G. Carle. "Open-Source MQTT Evaluation". In: *2021 IEEE 18th Annual Consumer Communications Networking Conference (CCNC)*.

[5] J. Benet. "IPFS - Content Addressed, Versioned, P2P File System". In: *CoRR* abs/1407.3561 (2014). URL: http://arxiv.org/abs/1407.3561.

[6] A. Broring, S. Schmid, C. Schindhelm, A. Khelil, S. Kabisch, D. Kramer, D. Le Phuoc, J. Mitic, D. Anicic, and E. Teniente. "Enabling IoT Ecosystems through Platform Interoperability". In: *IEEE Software* 34.1 (2017), pp. 54–61.

[7] B. Cohen. *Incentives Build Robustness in BitTorrent*. 2003.

[8] M. Cohen and A. Sundararajan. "Self-Regulation and Innovation in the Peer-to-Peer Sharing Economy". In: *University of Chicago Law Review Online* 82.1 (2015).

[9] *Decentralized Finance (DeFi)*. https://ethereum.org/en/defi/. [Online: Accessed 11-Feb-2021].

[10] *Decentralized Identifiers*. [Online]: https://www.w3.org/TR/did-core/. [Accessed: 15-Feb-2021].

[11] A. Dixit and A. Singh. *IoT Data Marketplace*. https://github.com/akankshadixit/IoTMarketplaces. 2021.

[12] *Electron*. https://electron.net. [Accessed: 11-Feb-2021].

[13] U. Habiba, R Masood, M. A. Shibli, and M. A. Niazi. "Cloud identity management security issues & solutions: a taxonomy". In: *Complex Adaptive Systems Modeling* 2.1 (2014).

[14] *Internet of Things (IoT) solutions*. [Online]. Available: https://aws.amazon.com/marketplace/solutions/IoT. [Accessed: 04-March-2021].

[15] W. Karl and A. Gervais. "Do you Need a Blockchain?" In: *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*. 2018.

[16] L. Xiong and L. Liu. "PeerTrust: supporting reputation-based trust for peer-to-peer electronic communities". In: *IEEE Transactions on Knowledge and Data Engineering* 16.7 (2004).

[17] Protocol Labs. *Filecoin*. https://filecoin.io. [Accessed: 11-Feb-2021].

[18] *LBRY*. https://lbry.com. [Accessed: 12-Feb-2021].

[19] J. Li, A. Grintsvayg, J. Kauffman, and C. Fleming. "LBRY: A Blockchain-Based Decentralized Digital Content Marketplace". In: *2020 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*. 2020.

[20] S. J. Liebowitz. *Re-thinking the Network Economy: The True Forces that Drive the Digital Marketplace*. AMACOM, 2002.

[21] *Medicalchain*. https://medicalchain.com/en/. [Accessed: 11-Feb-2021].

[22] P. Missier, S. Bajoudah, A. Capossele, A. Gaglione, and M. Nati. "Mind My Value: A Decentralized Infrastructure for Fair and Trusted IoT Data Trading". In: *Proceedings of the Seventh International Conference on the Internet of Things*. 2017.

[23] H. Niavis, N. Papadis, V. Reddy, H. Rao, and L. Tassiulas. "A Blockchain-based Decentralized Data Sharing Infrastructure for Off-grid Networking". In: *IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. 2020.

[24] *Open Bazar*. https://openbazaar.org. [Accessed: 12-Feb-2021].

[25] *OrbitDB*. https://orbitdb.org. [Accessed: 11-Feb-2021].

[26] *Origin*. https://www.originprotocol.com/en. [Accessed: 12-Feb-2021].

[27] G. S. Ramachandran, R. Radhakrishnan, and B. Krishnamachari. "Towards a Decentralized Data Marketplace for Smart Cities". In: *2018 IEEE International Smart Cities Conference (ISC2)*. 2018.

[28] R. Schollmeier. "A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications". In: *Proceedings First International Conference on Peer-to-Peer Computing*. 2001.

[29] D. Shi, J. Yin, Z. Wu, and J. Dong. "A Peer-to-Peer Approach to Large-Scale Content-Based Publish-Subscribe". In: *2006 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology Workshops*. 2006.

[30] *Sia*. https://sia.tech. [Accessed: 11-Feb-2021].

[31] *Storj*. https://storj.io. [Accessed: 11-Feb-2021].

[32] P. Thakkar and S. Natarajan. *Scaling Hyperledger Fabric Using Pipelined Execution and Sparse Peers*. 2021. arXiv: 2003.05113 [cs.DC].

[33] *The data exchange for advanced mobility and infrastructure*. [Online]: https://terbine.com. [Accessed: 04-March-2021].

[34] *To monetize and acquire IoT data*. [Online]: https://www.dawex.com/en/monetization-data-iot/. [Accessed: 04-March-2021].

[35] A. Tobin and D. Reed. "The Inevitable Rise of Self-Sovereign Identity". In: *The Sovrin Foundation*. 2016.

[36] Y. Xu, P. Ahokangas, S. Yrjölä, and T. Koivumäki. "The fifth archetype of electricity market: the blockchain marketplace". In: *Wireless Networks* (July 2019).