Autonomy, Efficiency, Privacy and Traceability in Blockchain-enabled IoT Data Marketplace

Author:
Gupta, Pooja

Publication Date:
2023

DOI:
https://doi.org/10.26190/unsworks/24905

Autonomy, Efficiency, Privacy and Traceability in Blockchain-enabled IoT Data Marketplace

Pooja Gupta

A thesis in fulfilment of the requirements for the degree of

Doctor of Philosophy

School of Computer Science and Engineering

Faculty of Engineering

The University of New South Wales

December 2022

THE UNIVERSITY OF NEW SOUTH WALES
Thesis/Dissertation Sheet

Surname or Family name: **Gupta**

First name: **Pooja** Other name/s: **Othernames**

Abbreviation for degree as given in the University calendar: **PhD**

School: **School of Computer Science and Engineering** Faculty: **Faculty of Engineering**

Title: Autonomy, Efficiency, Privacy and Traceability in Blockchain-enabled IoT Data Marketplace

Abstract

Personal data generated from IoT devices is a new economic asset that individuals can trade to generate revenue on the emerging data marketplaces. Blockchain technology can disrupt the data marketplace and make trading more democratic, trustworthy, transparent and secure. Nevertheless, the adoption of blockchain to create an IoT data marketplace requires consideration of autonomy and efficiency, privacy, and traceability.

Conventional centralized approaches are built around a trusted third party that conducts and controls all management operations such as managing contracts, pricing, billing, reputation mechanisms etc, raising concern that providers lose control over their data. To tackle this issue, an efficient, autonomous and fully-functional marketplace system is needed, with no trusted third party involved in operational tasks. Moreover, an inefficient allocation of buyers' demands on battery-operated IoT devices poses a challenge for providers to serve multiple buyers' demands simultaneously in real-time without disrupting their SLAs (service level agreements). Furthermore, a poor privacy decision to make personal data accessible to unknown or arbitrary buyers may have adverse consequences and privacy violations for providers. Lastly, a buyer could buy data from one marketplace and without the knowledge of the provider, resell bought data to users registered in other marketplaces. This may either lead to monetary loss or privacy violation for the provider. To address such issues, a data ownership traceability mechanism is essential that can track the change in ownership of data due to its trading within and across marketplace systems. However, data ownership traceability is hard because of ownership ambiguity, undisclosed reselling, and dispersal of ownership across multiple marketplaces.

This thesis makes the following novel contributions. First, we propose an autonomous and efficient IoT data marketplace, MartChain, offering key mechanisms for a marketplace leveraging smart contracts to record agreement details, participant ratings, and data prices in blockchain without involving any mediator. Second, MartChain is underpinned by an Energy-aware Demand Selection and Allocation (EDSA) mechanism for optimally selecting and allocating buyers' demands on provider's IoT devices while satisfying the battery, quality and allocation constraints. EDSA maximizes the revenue of the provider while meeting the buyers' requirements and ensuring the completion of the selected demands without any interruptions. The proof-of-concept implementation on the Ethereum blockchain shows that our approach is viable and benefits the provider and buyer by creating an autonomous and efficient real-time data trading model.

Next, we propose KYBChain, a Know-Your-Buyer in the privacy-aware decentralized IoT data marketplace that performs a multi-faceted assessment of various characteristics of buyers and evaluates their privacy rating. Privacy rating empowers providers to make privacy-aware informed decisions about data sharing. Quantitative analysis to evaluate the utility of privacy rating demonstrates that the use of privacy rating by the providers results in a decrease of data leakage risk and generated revenue, correlating with the classical risk-utility trade-off. Evaluation results of KYBChain on Ethereum reveal that the overheads in terms of gas consumption, throughput and latency introduced by our privacy rating mechanism compared to a marketplace that does not incorporate a privacy rating system are insignificant relative to its privacy gains.

Finally, we propose TrailChain which generates a trusted trade trail for tracking the data ownership spanning multiple decentralized marketplaces. Our solution includes mechanisms for detecting any unauthorized data reselling to prevent privacy violations and a fair resell payment sharing scheme to distribute payment among data owners for authorized reselling. We performed qualitative and quantitative evaluations to demonstrate the effectiveness of TrailChain in tracking data ownership using four private Ethereum networks. Qualitative security analysis demonstrates that TrailChain is resilient against several malicious activities and security attacks. Simulations show that our method detects undisclosed reselling within the same marketplace and across different marketplaces. Besides, it also identifies whether the provider has authorized the reselling and fairly distributes the revenue among the data owners at marginal overhead.

Signature **Pooja Gupta** Witness Date **11 May, 2023**

FOR OFFICE USE ONLY	Date of completion of requirements for Award

DECLARATIONS

PUBLICATIONS STATEMENT

☑ The candidate has declared that **their thesis has publications - either published or submitted for publication - incorporated into it in lieu of a Chapter/s. Details of these publications are provided below.**.

Publication Details #1

Full Title:	A Decentralized IoT Data Marketplace
Authors:	Pooja Gupta, Salil S. Kanhere and Raja Jurdak
Journal or Book Name:	Proceedings of 3rd Symposium on Distributed Ledger Technology (SDLT)
Volume/Page Numbers:	
Date Accepted/Published:	
Status:	published
The Candidate's Contribution to the Work:	Conceived and designed the study, acquired and analyzed data from experiments; contributed data or analysis tools; and wrote the manuscript.
Location of the work in the thesis and/or how the work is incorporated in the thesis:	The work is incorporated in the chapter 3 in the thesis.

Publication Details #2

Full Title:	Energy-aware Demand Selection and Allocation for Real-time IoT Data Trading,
Authors:	Pooja Gupta, Volkan Dedeoglu, Kamran Najeebullah, Salil S. Kanhere, Raja Jurdak
Journal or Book Name:	IEEE International Conference on Smart Computing
Volume/Page Numbers:	
Date Accepted/Published:	
Status:	published
The Candidate's Contribution to the Work:	Conceived and designed the study, acquired and analyzed data from experiments; contributed data or analysis tools; and wrote the manuscript.
Location of the work in the thesis and/or how the work is incorporated in the thesis:	The work is incorporated in the chapter 3 in the thesis.

Publication Details #3

Full Title:	TrailChain: Traceability of Data Ownership across Blockchain-Enabled Multiple Marketplaces
Authors:	Pooja Gupta, Volkan Dedeoglu, Salil S. Kanhere, Raja Jurdak
Journal or Book Name:	Journal of Network and Computer Applications
Volume/Page Numbers:	
Date Accepted/Published:	
Status:	published
The Candidate's Contribution to the Work:	Conceived and designed the study, acquired and analyzed data from experiments; contributed data or analysis tools; and wrote the manuscript.
Location of the work in the thesis and/or how the work is incorporated in the thesis:	The work is incorporated as a chapter 5 in the thesis.

Publication Details #4

Full Title:	Blockchain Applications for IoT Data Marketplace
Authors:	Pooja Gupta
Journal or Book Name:	Blockchain for CyberPhysical Systems
Volume/Page Numbers:	
Date Accepted/Published:	
Status:	published
The Candidate's Contribution to the Work:	Conceived and designed the study, acquired and analyzed data from experiments; contributed data or analysis tools; and wrote the manuscript.
Location of the work in the thesis and/or how the work is incorporated in the thesis:	The work is incorporated in the chapter 3 in the thesis.

Candidate's Declaration

✓ **I confirm that where I have used a publication in lieu of a chapter, the listed publication(s) above meet(s) the requirements to be included in the thesis. I also declare that I have complied with the Thesis Examination Procedure.**

Abstract

Personal data generated from IoT devices is a new economic asset that individuals can trade to generate revenue on the emerging data marketplaces. Blockchain technology can disrupt the data marketplace and make trading more democratic, trustworthy, transparent and secure. Nevertheless, the adoption of blockchain to create an IoT data marketplace requires consideration of autonomy and efficiency, privacy, and traceability.

Conventional centralized approaches are built around a trusted third party that conducts and controls all management operations such as managing contracts, pricing, billing, reputation mechanisms etc, raising concern that providers lose control over their data. To tackle this issue, an efficient, autonomous and fully-functional marketplace system is needed, with no trusted third party involved in operational tasks. Moreover, an inefficient allocation of buyers' demands on battery-operated IoT devices poses a challenge for providers to serve multiple buyers' demands simultaneously in real-time without disrupting their SLAs (service level agreements). Furthermore, a poor privacy decision to make personal data accessible to unknown or arbitrary buyers may have adverse consequences and privacy violations for providers. Lastly, a buyer could buy data from one marketplace and without the knowledge of the provider, resell bought data to users registered in other marketplaces. This may either lead to monetary loss or privacy violation for the provider. To address such issues, a data ownership traceability mechanism is essential that can track the change in ownership of data due to its trading within and across marketplace systems. However, data ownership traceability is hard because of ownership ambiguity, undisclosed reselling, and dispersal of ownership across multiple marketplaces.

This thesis makes the following novel contributions. First, we propose an autonomous and efficient IoT data marketplace, MartChain, offering key mechanisms for a marketplace leveraging smart contracts to record agreement details, participant ratings, and data prices in blockchain without involving any mediator. Second, MartChain is underpinned by an Energy-aware Demand Selection and Allocation (EDSA) mechanism for optimally selecting and allocating buyers' demands on provider's IoT devices while satisfying the battery, quality and allocation constraints. EDSA maximizes the revenue of the provider while meeting the buyers' requirements and ensuring the completion of the selected demands without any interruptions. The proof-of-concept implementation on the Ethereum blockchain shows that our approach is viable and benefits the provider and buyer by creating an autonomous and efficient real-time data trading model.

Next, we propose KYBChain, a Know-Your-Buyer in the privacy-aware decentralized IoT data marketplace that performs a multi-faceted assessment of various characteristics of buyers and evaluates their privacy rating. Privacy rating empowers providers to make privacy-aware informed decisions about data sharing. Quantitative analysis to evaluate the utility of privacy rating demonstrates that the use of privacy rating by the providers results in a decrease of data leakage risk and generated revenue, correlating with the classical risk-utility trade-off. Evaluation results of KYBChain on Ethereum reveal that the overheads in terms of gas consumption, throughput and latency introduced by our privacy rating mechanism compared to a marketplace that does not incorporate a privacy rating system are insignificant relative to its privacy gains.

Finally, we propose TrailChain which generates a trusted trade trail for tracking the data ownership spanning multiple decentralized marketplaces. Our solution includes mechanisms for detecting any unauthorized data reselling to prevent privacy violations and a fair resell payment sharing scheme to distribute payment among data owners for authorized reselling. We performed qualitative and quantitative evaluations to demonstrate the effectiveness of TrailChain in tracking data ownership using four private Ethereum networks. Qualitative security analysis demonstrates that TrailChain is resilient against several malicious activities and security attacks. Simulations show that our method detects undisclosed reselling within the same marketplace and across different marketplaces. Besides, it also identifies whether the provider has authorized the reselling and fairly distributes the revenue among the data owners at marginal overhead.

Acknowledgement

Undertaking this PhD has been a life-changing experience for me, and it would not have been possible without the support and guidance I received from numerous people.

Prima facie, I am grateful to Almighty for giving me the strength, ability, and opportunity to undertake this research work.

I wish to express my deepest and warmest gratitude to my supervisors, Professor Salil S. Kanhere, Professor Raja Jurdak and Dr Volkan Dedeoglu, for their continuous support, encouragement, patience and kindness during my PhD study. I also thank them for their immense knowledge, invaluable experience and insightful comments, which steered me to widen my research from various perspectives. Many thanks to Dr Volkan Dedeoglu for his mentorship. I will be forever indebted to them for the knowledge and technical skills I gained from them throughout. Working under their guidance and supervision has been a great privilege and honour.

I extend my heartfelt thanks to my beloved husband, Mohit. Without his tremendous understanding, unfailing support and numerous sacrifices over the past few years, it would have been impossible for me to complete this journey. I am very much thankful to my parents, Archna and Pradeep Gupta, for their love and prayers that sustained me this far. Special thanks to my cat, Tubby, for all the entertainment and emotional support.

Finally, I am grateful to the University of New South Wales for the scholarship opportunity to undertake my studies. I would also like to thank my colleagues at the University of New South Wales, including Dr Ali Dorri, Dr Guntur Dharma Putra, Dr Sidra Malik, and Samuel Karumba, for their technical support.

To all who are not mentioned but in one way or another helped in the completion of this study, thank you.

Pooja Gupta

Contents

List of Figures

List of Tables

Abbreviations

ABI	Application Binary Interface
ACL	Access Control List
AI	Articifical Inteliigence
BFT	Byzantine Fault Tolerance
CIC	Computationally Intensive Contracts
CPA	Certified Public Accountants
Dapp	Decentralized Application
DAPS	Double Authentication Preventing Signature
DFS	Distributed File Storage
DHT	Distributed Hash Table
DLP	Data Leakage Detection Or Prevention Systems
DORP	Data Ownership Registration Protocol
DRM	Digital Rights Management
DVP	Data Verification Process
EDSA	Energy-aware Demand Selection And Allocation
HCD	High Critical Data
ICO	Initial Coin Offering
ILP	Integer Linear Program
IoT	Internet Of Things
IPFS	Interplanetary File System
LCB	Least Significant Bit

LCD	Low Critical Data
MAM	Masked Authenticated Messaging
MKP-AR	Multiple Knapsack Problem With Assignment Restriction
ML	Machine Learning
MQTT	Message Queuing Telemetry Transport
NFT	Non-Fungible Tokens
P2P	Peer-to-peer
PDS	Personal Data Stores
PET	Privacy Enhancing Technology
PKI	Public Key Infrastructure
POA	Proof Of Authencity
POS	Proof-of-stake
POW	Proof-of-work
PSL	Payment Share List
RPS	Resell Payment Share
SGX	Software Guard Extensions
SLA	Service Level Agreements
SOC	System And Organization Controls
SP	Service Providers
TEE	Trusted Execution Environment
TLS	Transport Layer Security
TLV	Trade Lineage Verification
TTL	Trade Trail List
TTP	Trusted Third Party
VAS	Value Added Service
WTA	Willingness To Accept
WTP	Willingness To Pay
ZKP	Zero-knowledge Proof

Publications

Thesis Publications

The work conducted under this thesis has led to the following publications (listed in chronological order)

- **Pooja Gupta**, Salil S. Kanhere and Raja Jurdak, "A Decentralized IoT Data Marketplace", in Proceedings of 3rd Symposium on Distributed Ledger Technology (SDLT), Gold Coast, November 2018 DOI: `https://doi.org/10.48550/arXiv.1906.01799`
- **Pooja Gupta**, Volkan Dedeoglu, Kamran Najeebullah, Salil S. Kanhere, Raja Jurdak, "Energy-aware Demand Selection and Allocation for Real-time IoT Data Trading," In proceedings of the 6th IEEE International Conference on Smart Computing (Smart Comp), Bologna, Italy, September 2020 DOI: `https://doi.org/10.1109/SMARTCOMP50058.2020.00038`
- **Pooja Gupta**, Volkan Dedeoglu, Salil S. Kanhere, Raja Jurdak, "Towards a blockchain powered IoT data marketplace", In proceedings of IEEE International Conference on COMmunication Systems & NETworkS (COMSNETS), January, 2021 DOI: `https://doi.org/10.1109/COMSNETS51098.2021.9352865`
- **Pooja Gupta**, Volkan Dedeoglu, Salil S. Kanhere, Raja Jurdak, "Data Reselling in IoT Data Marketplace", In proceedings of the 7th ACM Celebration of Women in Computing: womENcourage 2020 conference, Baku, Azerbaijan, September, 2020 Available Online: `http://womencourage.acm.org/2020/wp-content/uploads/2020/07/womENcourage_2020_paper_15.pdf`
- **Pooja Gupta**, "Blockchain Applications for IoT Data Marketplace", Blockchain for CyberPhysical Systems, ARTECH House, 2020 Available Online: `https://us.artechhouse.com/Blockchain-for-Cyberphysical-Systems-P2108.aspx`
- **Pooja Gupta**, Volkan Dedeoglu, Salil S. Kanhere, Raja Jurdak, "TrailChain: Traceability of Data Ownership across Blockchain-Enabled Multiple Marketplaces", Journal of Network and Computer Applications, 203, 103389 , 2022 DOI: `https://doi.org/10.1016/j.jnca.2022.103389`

Chapter 1

Introduction

"Data is the new oil. It's valuable, but if unrefined, it cannot be used. It has to be changed into gas, plastic, chemicals, etc. to create a valuable entity that drives profitable activity; so must data be broken down and analyzed for it to have value." – Clive Humby, 2006

The above statement by Clive Humby [1] that data is the new oil has often been cited. In some ways, this expression accurately reflects how data is an essential resource of this century, similar to oil a century ago. Businesses are investing enormous resources to accumulate and take control of data, and with its possession, they can transform society and the economy. However, this statement fails to capture the essence of the data. Unlike oil, data is not a finite resource, it can be reused, and its value depends on the eye of the beholder. Therefore, a better analogy is to think of data as an asset.

1.1 Data as an Asset

In 2011, the World Economic Forum [2] described data as a new asset in the digital economy. One of the driving forces behind the digital economy is data generated from the Internet of Things (IoT) devices. With the continued adoption of IoT, more 'things' are becoming connected to the Internet, such as wearable devices, medical implants, cars, etc. The IDC forecast [3] estimates that over 41.6 billion devices will be connected by 2025, collectively producing 73.1 ZB (zettabytes) of data. This voluminous data may contain sensitive information about our identity, social interactions, habit, locations etc. Collecting, storing and analyzing these data will pose enormous opportunities in different application domains [4–8] including agriculture, automotive, home automation, health care, insurance services, personal fitness, smart cities, and many more. Organizations collect and use IoT data to improve and optimize operations [9]. For example, ride-booking services can use customers' smartphone data to monitor traffic conditions in real time, adjust pricing based on demand and manage vehicle supply. Others could employ IoT data as an integral part of their product/service offering [8]. For example, health insurance providers can use health data from fitness or wearable devices to develop customer-centric propositions that incentivize healthy lifestyles. IoT data can also improve the effectiveness of critical public services [10]. For instance, municipal governments can collect sensing data from smart cars to predict the development of potholes for road maintenance. Researchers can use data generated from smart devices and wearables to develop models that can detect and prevent infections [4]. For example, with the help of health parameters collected in real-time from mobile phone sensors, COVID-19 positive and suspected cases can be tracked and quarantined. In short, data is a valuable resource of the 21st century touching all aspects of society.

1.2 The IoT ecosystem - where we stand today

As highlighted in the above examples, acquiring the correct set of IoT data at the correct time and context can help businesses, governments, service providers, developers and other data consumers to make more informed decisions. However, one significant challenge data consumers face is the considerable time and effort required to access high-quality IoT data generated by disparate data sources. There are several models under which these data sources could be owned and operated [11]. They may a) be wholly owned and operated by a city government; b) they may be deployed and operated/managed by a single organization for each application; or c) they may belong to individuals who are normal citizens, producing various types of data streams regularly from their sensor-equipped devices. However, all these three options are not always available in many use cases. For example- (a) and (b) do not make sense for the wearable. Therefore, our interest is in the data collected by the sensors built into the heterogeneous smart devices owned by individuals because it is intuitively the most scalable and effective way to collect large datasets. The most common example is the smartphone sensors already in the hands of 6.64 billion [12] people worldwide. A smartphone is typically equipped with a rich set of embedded sensors [13], such as an accelerometer, digital compass, gyroscope, GPS, microphone, camera, etc. Moreover, IoT devices are an integral part of individuals' lives, from light bulbs to coffee makers, headphones, TVs, and many more. By 2030, it is expected that each individual will possess about 15 connected devices [14]. However, with the amount of data generated by billions of IoT devices, it is challenging for data consumers to channel, collect, manage, and find needed data from all these data sources.

IoT comprises low-power devices [15] with necessary sensors to fulfil primary usage. For example, a smart sprinkler may only be activated if the soil moisture level in the garden goes below a certain level. Typically, data generated by these IoT devices are managed and stored in a cloud by application or service providers (SP). SP process and analyze the collected data to provide a promised service, and after his analysis, data is either thrown away or amassed in private silos. Such unused and hidden data are known as dark data, which accounts for 99% of the collected data [16]. Dark data hide a considerable amount of knowledge and insight that could be used to build valuable services. To realize the full value of data, these dark data residing in monolithic and isolated data silos must be combined from multiple IoT solutions and made easily accessible or shareable across services. However, in the current ecosystem, SP allows data owners to access their data or summaries through some application or web. Data owners have no option but to trust the SP and rely on their promises of resilience, availability, and security. Moreover, these data often contain sensitive private information that SPs can use to profile data owners violating their privacy and making profits through unsolicited marketing or targeted advertisements [17,18]. Besides, SP could also exploit and make users' data widely available

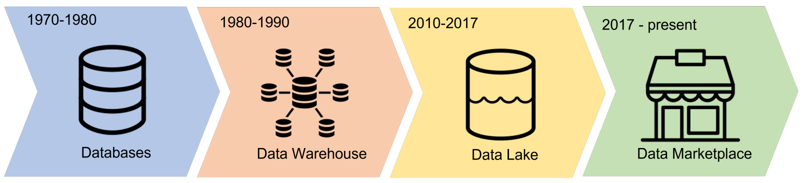


Figure 1.1: Evolution of data management system

to anyone without the user's explicit consent, for example, in the Facebook Cambridge Analytica scandal [19]. In 2018, Cambridge Analytica misused the API, originally designed to allow a third-party app to access the personality profile of participating users, to collect information on 50 million Facebook profiles without the consent of the users. Such illicit, gathered private data are later used to create personalized psychology profiles for political purposes. Hence, in the current IoT ecosystem, data owners have limited possibilities to control and monetize their data.

The aforementioned limitations in the current IoT ecosystem necessitate a mechanism that empowers data owners with fine-granular access control, allowing them to decide what data they wish to share with whom and ensure data ownership. Furthermore, there should be a way for data owners to demand fair compensation for the data they share with data consumers. Finally, the data owners must be able to reject any request for data sharing they deem unacceptable. These requirements lead us to conclude that a marketplace for IoT data that aims to ensure ownership, control and fairness is viable.

1.3 The rise of Data Marketplace

This section will discuss the evolution of data management systems traditionally used to manage data silos, their limitations and how the IoT data marketplace is a promising solution for data consumers and owners. The evolution of the data management systems is depicted in Fig. 1.1. Between 1970-1980, databases were developed to store many structured data types such as names, dates, addresses etc., enabling users to interact with the data. However, databases are expensive and proprietary and are often considered unsuitable for meeting IoT data requirements [20] due to the dynamic and vast nature

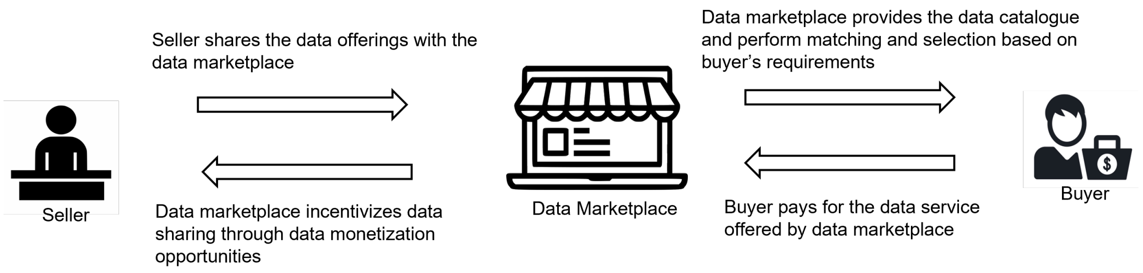

Seller shares the data offerings with the data marketplace
Data marketplace incentivizes data sharing through data monetization opportunities
Seller
Data Marketplace
Data marketplace provides the data catalogue and perform matching and selection based on buyer's requirements
Buyer pays for the data service offered by data marketplace
Buyer

Figure 1.2: Data marketplace facilitating trade between seller and buyer

of IoT, heterogeneous data sources, size and scale, etc. Subsequently, the concept of data warehouses developed between 1980-1990 to accumulate data from a wide range of sources and transform it into a structured format for processing and storing purposes. However, data warehouses have a rigid and inflexible architecture that makes adding a vast amount of unstructured data from new IoT devices time-consuming. In 2011, data lakes were introduced to tackle the limitations of data warehouses. A data lake is a large storage repository that holds large volumes of raw data in its original format. However, data lakes do not offer a way to organize and monetize data. In 2017, the notion of an IoT data marketplace attracted increasing attention as it enables (i) data consumers to navigate, integrate, and analyze data produced from a plethora of IoT devices owned by individuals to generate timely insights that help in effective decision-making, (ii) data owners to control, monetize, and distribute data generated from their IoT devices with data consumers. Data marketplace addresses the issue of data silos and facilitates the data trade between data owners and data consumers. A data marketplace consists of two main participants: seller and buyer. A seller is an entity that sells data generated from his IoT devices in return for incentives. We will use seller and provider interchangeably in the thesis. A buyer is an entity interested in purchasing a particular type of data. For example, application developers, researchers, businesses, government, etc. A data marketplace, as depicted in Fig. 1.2, in general, facilitates all the activities of selling and buying, such as enabling the seller to list his data offerings and the buyer to send a query and discover data as per requirement, pricing the data, negotiating, managing and enforcing the data contracts, transferring data and making payment settlement.

IoT Data Marketplace and Significance of Decentralization

This section will discuss the marketplace's fundamental responsibilities and different approaches to realize these responsibilities and their limitation.

A data marketplace is an ecosystem that facilitates data trading between sellers and buyers by enabling governance. A marketplace offers services to deliver the main functions of the trading process, including data discovery, price model management, payment, data contract management and data transfer [21]. Data marketplace governance refers to the management and regulatory framework for decision-making that covers how the market operates and functions. Marketplace governance includes: deciding on the data pricing, supervising terms and conditions of trade, monitoring and enforcing agreements and contracts, defining rules and restrictions around data sharing for sellers and around access right for the buyers, ensuring the method for data collection and transfer, and so forth. Effective and efficient governance can be achieved by enabling transparency and accountability [22]. Transparency ensures that relevant information is available to the participants so they can know about the decision made by the marketplace. For example, sellers and buyers have access to pricing information, knowledge about trade transactions, usage policy, terms and conditions of trade, etc. Accountability ensures that any ambiguity in the trade process should be resolved based on a common agreement by all the participants, and liable participants should be held responsible for their actions. Without transparency and accountability, participants will lack trust in the marketplace [23]. The primary responsibility of the data marketplace is to help its participants build trust by providing governance to negotiate and sign a contract with an untrusted entity with whom they have no prior relationship. There are two approaches to designing a data marketplace, as depicted in Fig. 1.3.

Centralized approach - In this approach, a central trusted third party (TTP) handles the governance to ensure the proper operation of the marketplace. TTP oversees and controls every aspect of the trade, from offer listings to data pricing, receiving seller's raw

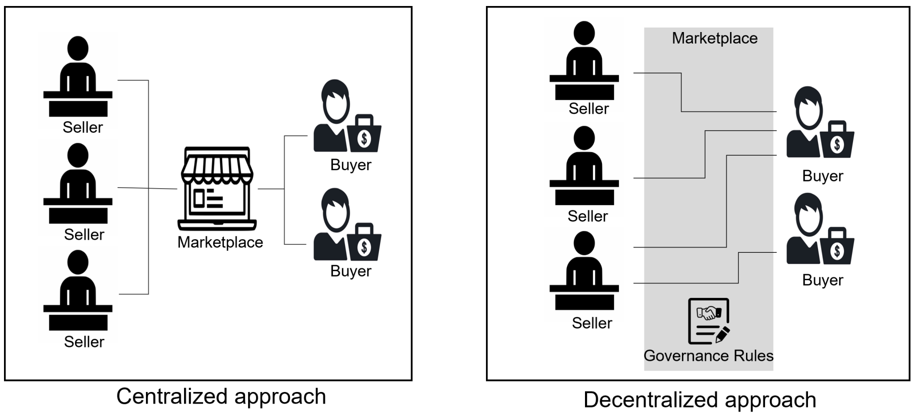


Figure 1.3: Centralized and decentralized approaches to implement marketplace

data, processing and storing it, transacting the trade, data dissemination, payment and settlements. The centralized governance of the TTP results in several limitations.

- Limited accountability: TTP has a central monopoly as they have exclusive ownership of all marketplaces operations. TTP can exploit its position, set the price and operation rules, and enforce that to the participants. All participants must comply with the TTP's terms, which can change anytime. Besides, TTP can collude with dishonest participants and cheat, leading to unfair trades.
- Limited privacy - TTP has complete visibility on the trade activity and history of the seller and buyer and the associated data. TTP does not guarantee that it will protect participants' information from misuse, interference and loss nor disclose trade details to unauthorized parties.
- Limited transparency: Limited information about TTP operations and activities are visible to the participants. Providers often have limited control over their data as they do not know whether it is altered, where it resides, etc. Similarly, buyers are often unaware of hidden overhead costs, provided with biased terms and conditions, misinformed about the provider's usage policy, etc.

With billions of connected IoT devices, centralized governance can be a bottleneck. Early works [21, 24] based on this approach rely on centralized systems such as cloud computing as an underlying technology. However, such centralized systems suffer from known issues such as scalability, expensive infrastructure, single point of failure, and low security, which make them unsuitable for managing the enormous volume of data generated by IoT devices. Additionally, traditional database techniques may not be suitable for designing IoT data marketplaces that need to scale to effectively store and manage data. As the amount of data stored in a centralized database grows, it can become increasingly difficult to efficiently manage and process the data, leading to slower response times and potential latency issues for users. Moreover, the centralized nature of traditional databases makes them vulnerable to cyber attacks [25], leading to data loss, theft, and manipulation [26]. Therefore, a decentralized approach is necessary for IoT data marketplaces that can offer higher security, transparency, and immutability as discussed next.

Decentralized approach - A decentralized approach could solve the limitations of the aforementioned approaches. A decentralized marketplace transfers the control and responsibilities from a TTP to a distributed network and facilitates P2P interactions by bridging the trust gap. It enables decentralized governance wherein each participant agrees on a set of business rules that will govern the transactions and are each responsible for running the marketplace infrastructure. The rules and infrastructure provide a framework that makes it possible to set up automated agreements. Since all the participants take part in the marketplace operation, a single entity cannot have a monopoly, making the marketplace transparent.

In a decentralized marketplace, no single entity takes responsibility for the proper operation of the marketplace. As a result, accountability must come from the underlying technology. Essential requirements of the decentralized marketplace to ensure accountability are: (i) the participants must be held accountable for their actions, and (ii) the marketplace must be able to account for actions performed by the participants. Ensuring user accountability requires each participant to be linked to a legal entity that can hold them responsible for their actions. Accounting for participant actions requires the marketplace to capture the nature of the action, what has been done, when, and by whom and record the associated information. For instance, a record to prove that the user has signed a contract at a given time and agreed to its terms. The marketplace also needs to ensure that the historical records are maintained, protected against tampering, and retrievable for auditing. Moreover, all these operations must be verifiable by any marketplace participants at any time. Since a decentralized marketplace stores sensitive information that should not be accessible by everyone, thus, data privacy becomes a critical challenge.

As stated above, the requirements of decentralized governance are commensurate with the properties of blockchain technology, which was first introduced by Nakamoto in the Bitcoin cryptocurrency [27]. The following section highlights the benefits of using blockchain in designing a decentralized IoT data marketplace and its challenges.

Blockchain: An Enabler of Decentralized IoT Data Marketplace

A blockchain [28] is a consensus-based, peer-to-peer distributed network with a growing list of ordered records, called blocks, that are chained together using cryptographic properties of hash functions. Typically, blockchain is a decentralized, distributed and public digital ledger that is used to record transactions shared among the nodes in the network, making it more secure and less prone to a single point of failure. Consensus allows distributed users to reach an agreement by communicating directly, enabling disintermediation and ensuring that all copies of the distributed ledger are in the same state. A transaction is a basic communication primitive between participating nodes in a blockchain. Each block contains a cryptographic hash of the previous block, a timestamp, and transaction data. Utilizing the cryptographic hash functions, any attempt to change one transaction result could be detected by a mismatch; hence, blockchain provides the immutability property, which ensures resilience against modification or removal of the stored data. Furthermore, blockchain also provides a technology called smart contracts, which are pieces of executable code residing on the blockchain and automatically execute once specific conditions are met.

The multidimensional benefits of the blockchain in the IoT data marketplace [11, 29–31] are summarized in Fig. 1.4. A user in a blockchain network is known by a pseudonym,

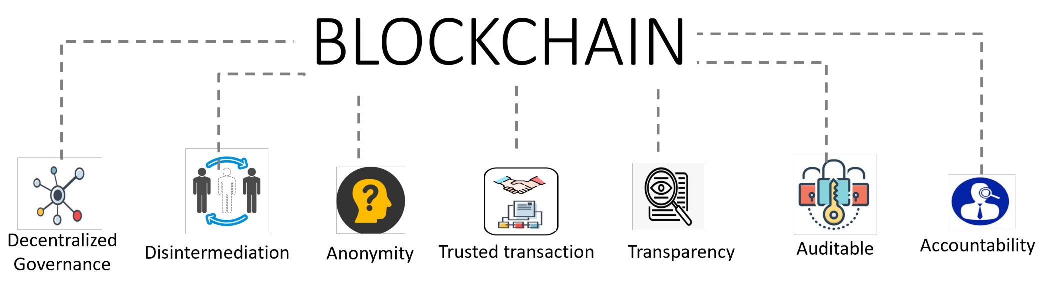


Figure 1.4: The fundamental advantages of the blockchain in data marketplace

which is its public key, instead of the real-world identity, enhancing user anonymity. Every time users transact IoT data, the corresponding transaction records are grouped into blocks. These blocks are added to the blockchain after users validate the information authenticity using consensus, which supports trust in every transaction. The fact that all the trade-related transactions are stored in the ledger makes it possible for users to verify the veracity of transactions. The existence of an encrypted, perpetual, and validated record of transactions via blockchain increases transparency and confidence in buyers and providers to exchange their sensitive IoT data. Users can employ smart contracts to encode the terms of an agreement in lines of code instead of legal language. These contracts self-execute when pre-determined conditions are met that automatically enforce the terms of the agreements and establish effective governance that all nodes in the network follow. Hence, using blockchain in the data marketplace enables decentralized governance, and disintermediation ensures user anonymity, increases transparency and trust in transactions, creates an immutable and auditable log of historical transactions, and promotes accountability of users' activities.

1.4 Thesis Motivation

In this section, we discuss the main challenges to exploit the useful features of blockchain technology in a decentralized IoT data marketplace, followed by specific contributions of this thesis in the next section.

Despite all the potential benefits, adopting blockchain to create an open IoT data marketplace is not straightforward and requires design consideration from different perspectives. In this research, we aim to design decentralized IoT data marketplaces of the future by considering design challenges from autonomy, efficiency, privacy and traceability aspects.

1.4.1 Autonomy and efficiency

An autonomous marketplace enables self-governance where participants can make trade decisions rather than being imposed by the marketplace. Such a marketplace establishes sellers' autonomy through increased transparency, i.e., by letting them know of their rights and allowing them to control who has access and to what data, how their data is bought, sold, shared etc. Buyers can exercise their autonomy by negotiating the data price and the terms and conditions under which data is shared. The adoption of blockchain in recent works to develop a decentralized marketplace provides specific functionality for data trading. For instance, in [29,30,32], blockchain records all the trade activities that improve transparency and accountability with verifiable logs. This approach facilitates autonomy but recording every interaction on the blockchain is inefficient. In [33–35], blockchain is used as an access control system to automatically manage data access based on the criteria set by sellers. This approach enables autonomy for sellers who can control their data access, but buyers have limited autonomy in negotiating and deciding the terms and conditions. Other approach [11,36,37] use blockchain to record the sellers' data offerings, and buyers can browse the ledger to find the proper data type based on their requirements. This approach facilitates the least level of autonomy allowing only data listing and querying. The current approaches lack complete autonomy that would allow participants to pursue their interests and impel them to behave responsibly to establish trust.

Moreover, blockchain is resource-consuming since all the entities involved must perform consensus or encryption algorithms and have to place a complete copy of the blockchain, which comes at the cost of scalability. On the contrary, IoT devices are resource-constrained with limited computational, memory and networking constraints. Even with lighter consensus mechanisms, having all IoT devices execute the associated algorithms is inefficient. Also, IoT devices do not always have the required storage space to retain copies of all transactions recorded in blockchain history.

To achieve autonomy and efficiency in a decentralized IoT data marketplace, we identify the following requirements:

- An open data marketplace is accessible to anyone to join and trade. Such a marketplace often spans the globe, with many participants. Therefore, scalability is a fundamental challenge for efficiency in the IoT data marketplace. With the ever-increasing number of IoT devices owned by many sellers, generating large amounts of IoT data would lead to an overwhelming number of transactions. Using one monolithic blockchain for storing all the marketplace-related transactions, such as all trade interactions, data offerings, and access policies on-chain, is expensive, inefficient and unscalable. The intrinsic trade-off between scalability and the need to maintain a decentralized and distributed architecture makes it challenging to use a fully decentralized public blockchain network like Bitcoin or Ethereum, as they may not be able to handle the transaction load.
- Ensuring non-repudiation in an autonomous marketplace without an intermediary is challenging. An autonomous marketplace enables self-governance where participants define the terms and conditions under which they trade the data. Non-repudiation ensures that they adhere to those obligations made under a contract. Consider an example where a buyer and seller agree to trade GPS data with specific sampling intervals, duration and quality at a certain price. However, ensuring that both parties adhere to these obligations without an intermediary is difficult. The seller may renounce the contract by not providing the GPS data for the specified requirement. Similarly, the buyer may not pay the agreed price for the purchased data. Such actions lead to agreement violations and impact the usability of the data marketplace.
- Another fundamental challenge in trading data-stream from IoT devices is its en-

ergy budget, which may limit the volume of data generated in real time. An IoT device's energy budget is subject to its battery capacity or, in some cases, to external sources such as harvesters. Without energy harvesting, IoT devices must optimize their operation and adapt to their environments to fulfil their primary usage. With energy harvesting, IoT devices have a dynamic energy budget subject to their physical context, namely, their current draw and perspective. Since IoT devices are designed to operate autonomously for a long duration and typically require little or no human interaction, they must be energy efficient. Hence to be viable in the data marketplace, an IoT device must serve its primary purpose without impacting the user's experience. Besides, it should also serve the secondary purpose of fulfilling buyers' data collection requirements based on the device's current residual energy and capabilities.

To realize the above requirements of autonomy and efficiency in a decentralized IoT data marketplace, we formulate and seek answers to the following key research questions:

RQ1: *How to address the scaling vs. decentralization trade-off to achieve autonomy and efficiency in decentralized IoT data marketplace?*

RQ2: *How to ensure non-repudiation among the participants who do not trust each other without an intermediary?*

RQ3: *How to efficiently utilize the energy of an IoT device to serve both primary and secondary purposes in the marketplace to deliver maximum utility within its energy constraints?*

1.4.2 Privacy

Privacy concerns in trading IoT data are high as it can reveal sensitive and confidential information about the users, such as their daily habits, locations and health. Blockchain provides pseudonymization using public-private key cryptography to protect the true identity of the participants. However, attackers can access the full history of all the trading transactions between the parties, making them vulnerable to linking or de-anonymization. Hence, privacy is another challenge in the blockchain-enabled IoT data marketplace. To address the issue of privacy, we identify the following requirements:

- A provider receives short-term economic gain by trading their IoT data in such a marketplace. As their engagement increases, trading data with a particular buyer over the longer term may affect their privacy. Moreover, data is considered an asset and requires protection even after being sold. Therefore, it becomes the buyer's responsibility to safeguard the provider's data privacy. Three characteristics of the buyer that raise the provider's privacy concerns and risks include (i) Non-compliance: a buyer may not have adopted or complied with the industry standard privacy practice and security measures to safeguard the provider's data; (ii) Data accumulation: large accumulation of data by the buyer can make them a target of cybercrime or a buyer acting in bad faith can misuse this data to infer highly personal information about the provider; (iii) Data-leak: a buyer may intentionally or accidentally share the provider's sensitive data with an unauthorized party without the provider's knowledge or consent. The provider is often unaware of all these risks when deciding to sell data to the buyer. Existing data marketplace ecosystems rely on the buyer's trust score that governs the provider's decision of whether or not to give data access to them. However, the trust score is a subjective metric that depends on his interactions, historical behaviour, experience, feedback or ratings submitted by providers. We identify the following three requirements to address the privacy issues from the provider's perspective. First, providing factual and relevant information about buyers is essential to empower providers to estimate the aforementioned privacy risks and make a more confident and informed decision about data disclosure to them. Second, even if providers have access to comprehensive and multifaceted information about buyers, it will take time and effort to digest all this information. Therefore, it is necessary to provide providers with an effective and simple way to ascertain the implications of selling data to the buyer on their privacy. Last, since the provider's decision of data disclosure depends on the buyer's information, it is vital that this information be accurate and timely and captures the buyer's dynamic activities in the marketplace. Therefore, to prevent manipulation and build trust, this information should be collated and managed in a decentralized, transparent, efficient, secure and automated manner.

To fulfil the above requirements of privacy in a decentralized IoT data marketplace, we formulate and seek answers to the following research questions:

RQ4: *What relevant information about buyers would empower providers to estimate their privacy risks and make a privacy-aware, informed decision about data disclosure?*

RQ5: *How can we convey buyers' comprehensive and multifaceted information simply and practically to the provider so that they can effectively ascertain the implications of selling their data to buyers on their privacy?*

RQ6: *How to ensure that buyers' information is accurate, not manipulated, and collated timely and automatically based on their dynamic activities in the decentralized marketplace?*

1.4.3 Traceability

The immutable feature of the blockchain makes it an effective tool for ownership verification since once an asset is listed on the blockchain, ownership cannot be altered or counterfeited unless the owner verifies the change. Blockchain benefits in tracking and tracing on-chain digital assets like bitcoin since it is minted, listed or transferred, but data stored off-chain is different. Due to the severe scalability and usability impact, recording enormous amounts of IoT data on the blockchain is not possible. To address this issue, one way is to record the hash on-chain that will direct the buyers to off-chain storage where IoT data is stored. However, a buyer can independently resell the purchased data off-chain to others without the provider's knowledge, making tracking and tracing data ownership challenging in the IoT data marketplace.

To accomplish the traceability of data ownership in a decentralized marketplace, some of the specific challenges are listed below:

- Data ownership is ambiguous, as buyers may claim ownership of the purchased data. Unlike traditional physical commodities that are owned by one specific entity, data ownership can include multiple parties as long as they can access the data and share it with others. Hence, it is difficult to keep track of ownership once data is sold. Furthermore, buyers will likely register with multiple marketplaces for monetary gain or alternative offerings for data purchases. A buyer could buy data from one marketplace and resell the purchase data in its original format or redistribute it as a value-added service (VAS) to users registered in other marketplaces. Current marketplace architectures are fragmented and dispersed and lack mechanisms to track data ownership spanning multiple marketplace systems.
- A data owner can explicitly consent to the buyer to resell his data (in the original format or VAS) to others to maximize his profit. This requires all entities involved in the reselling process to be treated fairly. Blockchain guarantees trade fairness, which means a provider should receive a payment if and only if a buyer acquires the expected data and vice-versa. While reselling fairness means that a reseller who buys data from a provider and obtains payment by reselling the purchased data to other users should fairly distribute the profit among all the data owners. However, a buyer can independently resell the bought data to other buyers and generate value from it without the data owner's knowledge. Such undisclosed reselling may lead to privacy violations when the data owner does not want to resell their data (unauthorized reselling) or monetary losses when the data owner permits it to be resold for economic interests (authorized reselling). Such undisclosed data resales should be automatically detected to enable reselling fairness.
- Since an honest buyer pays for the data, they should be ensured that the traded data is accurate, authentic, and sold with explicit consent. However, a malicious actor can perform various fraudulent activities, such as trying to impersonate IoT devices to sell fake or bogus data or tamper with the data sent by legitimate devices. Trading

such bogus or fake data impacts the marketplace's effectiveness and usability. This necessitates a mechanism that enables buyers to validate the authenticity of the purchased data by ensuring that it is generated from a legitimate device and has not been altered through the entire life cycle.

To address the aforementioned challenges in data ownership traceability, we formulate and seek answers to the following key research questions:

RQ7: *How to manage and track the data ownership spanning across multiple marketplace systems that are dispersed and fragmented?*

RQ8: *How to formulate the revenue sharing scheme to automatically and fairly distribute the revenue among all the data owners over authorized reselling of data and protect the profit of the data owners from malicious buyers that may resell without an authorization?*

RQ9: *How to ensure that only authentic and accurate data generated from a legitimate device is traded in the marketplace? How to enable buyers to validate the authenticity of the purchased data?*

Implementation issue

The complexity of smart contracts is another challenge in blockchain implementation because of its impact on cost and network speed measured in terms of latency and throughput. Since recording anything on the blockchain results in a data storage fee, deploying a smart contract incurs a transaction cost that depends on the size and the computational tasks it performs. Besides, each function call in the smart contract code also has an execution cost that depends on its space, time and transaction complexity. Space complexity includes both memory and ledger storage consumption. Time complexity typically reflects the complexity of the instructions or operations and their combinations. Transaction complexity depends on the number of arguments or volume of data in/out of a blockchain client. Furthermore, structural designs of smart contracts with complicated validation logic and more number of reads and writes from/to the ledger may increase the processing latency and reduce the throughput of the blockchain network, thus, impacting the overall performance. To address this concern, we formulate and seek answers to the following implementation question:

IQ: *Given the volume of marketplace-related events and frequency of trades, what are the major design considerations in architecting a blockchain solution for decentralized IoT data marketplace so that the associated overheads (in terms of gas consumption, latency and throughput etc.) should be minimal and not impact the performance of the platform?*

To this end, *this thesis contributes to the research in the adoption of blockchain technology to design a decentralized IoT data marketplace addressing challenges from autonomy, efficiency, privacy and traceability aspects* as stated above.

1.5 Thesis Contributions

Primarily, this thesis contributes to the adoption of blockchain technology in designing a decentralized IoT data marketplace to achieve autonomy, efficiency, privacy and traceability by addressing the challenges outlined in the previous section. The specific contributions of this thesis are outlined in the following subsections. Fig. 1.5 depicts the research objective and contributions of the thesis graphically.

In this thesis, we have presented our conceptual ideas in the application domain of personal IoT data. Despite this, our ideas can be generalised to other IoT data-sharing scenarios/application domains. The reason for choosing personal IoT data is two-fold: An increased incidence of amassing data without the knowledge of data owners and drawing inferences or illegally sharing it with third parties in recent years has led to the demand for user-controlled sharing where an individuals decides what data they wants to share, to who and for how long. Thus, addressing this demand for the democratization of all the untrusted participants by allowing them to interact with each other using blockchain technology could be apt in the current IoT ecosystem. Secondly, the related data, roles

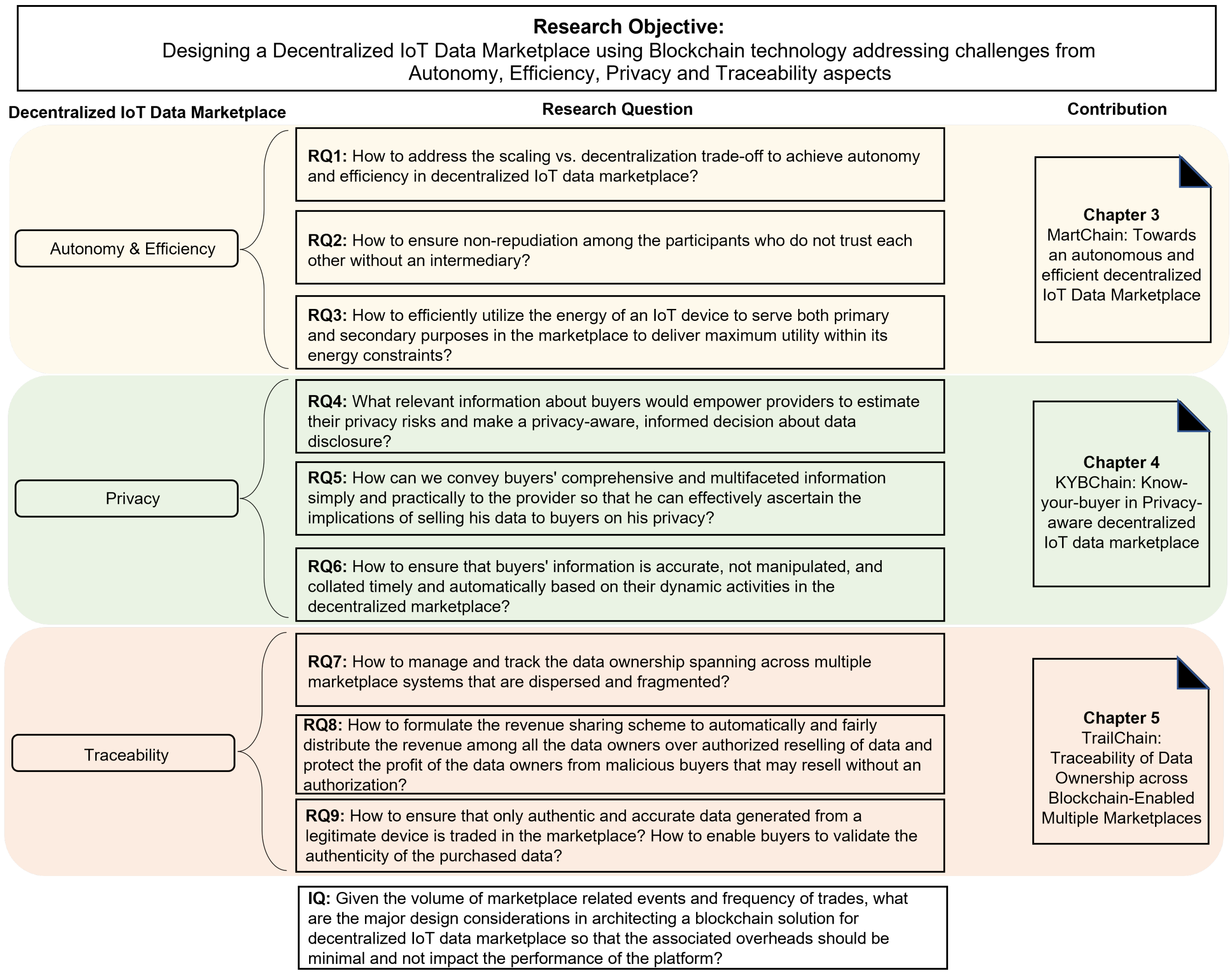

Research Objective:
Designing a Decentralized IoT Data Marketplace using Blockchain technology addressing challenges from Autonomy, Efficiency, Privacy and Traceability aspects
Decentralized IoT Data Marketplace
Research Question
Contribution
Autonomy & Efficiency
RQ1: How to address the scaling vs. decentralization trade-off to achieve autonomy and efficiency in decentralized IoT data marketplace?
RQ2: How to ensure non-repudiation among the participants who do not trust each other without an intermediary?
RQ3: How to efficiently utilize the energy of an IoT device to serve both primary and secondary purposes in the marketplace to deliver maximum utility within its energy constraints?
Chapter 3 MartChain: Towards an autonomous and efficient decentralized IoT Data Marketplace
Privacy
RQ4: What relevant information about buyers would empower providers to estimate their privacy risks and make a privacy-aware, informed decision about data disclosure?
RQ5: How can we convey buyers' comprehensive and multifaceted information simply and practically to the provider so that he can effectively ascertain the implications of selling his data to buyers on his privacy?
RQ6: How to ensure that buyers' information is accurate, not manipulated, and collated timely and automatically based on their dynamic activities in the decentralized marketplace?
Chapter 4 KYBChain: Know-your-buyer in Privacy-aware decentralized IoT data marketplace
Traceability
RQ7: How to manage and track the data ownership spanning across multiple marketplace systems that are dispersed and fragmented?
RQ8: How to formulate the revenue sharing scheme to automatically and fairly distribute the revenue among all the data owners over authorized reselling of data and protect the profit of the data owners from malicious buyers that may resell without an authorization?
RQ9: How to ensure that only authentic and accurate data generated from a legitimate device is traded in the marketplace? How to enable buyers to validate the authenticity of the purchased data?
Chapter 5 TrailChain: Traceability of Data Ownership across Blockchain-Enabled Multiple Marketplaces
IQ: Given the volume of marketplace related events and frequency of trades, what are the major design considerations in architecting a blockchain solution for decentralized IoT data marketplace so that the associated overheads should be minimal and not impact the performance of the platform?

Figure 1.5: Research objective and contributions

and participants in a marketplace trading personal IoT data generated by individuals are easier for the readers to understand from a user's perspective.

1.5.1 MartChain: Towards an autonomous and efficient decentralized IoT Data Marketplace

In this contribution, we propose a blockchain-based data marketplace known as MartChain for trading IoT data in real-time generated by resource-constrained IoT devices. To address the scaling vs decentralization trade-off, we propose to use geographically distributed multiple facilitators, to which participants delegate the responsibility of data offerings/demands listings and search and discovery (**RQ1**). We develop MartChain as a 2-tier marketplace framework where tier 1 comprises facilitators, and tier 2 comprises participants, including buyers, providers and their devices. We propose to use a public blockchain network in tier 2 that realizes other essential operational components of the marketplace, such as contract management and execution, data pricing, payment and settlement, and reputation mechanisms, leveraging smart contracts to enable autonomous IoT data trading. These contracts collectively ensure the integrity of an agreement, non-repudiation of the parties and guarantee that the participants' behaviours automatically conform to the terms of the agreements (**RQ2**). Due to the resource-constrained nature of IoT devices, an inefficient allocation of buyers' demands on such devices (i.e., secondary usage) poses a challenge for a provider to simultaneously serve multiple buyers' demands in real time without disrupting their SLAs (service level agreements). MartChain is underpinned with an Energy-aware Demand Selection and Allocation (EDSA) mechanism for optimally selecting and allocating buyers' demands on the provider's IoT devices (**RQ3**). We present a proof-of-concept implementation in Ethereum to demonstrate the feasibility of the framework (**IQ**). We investigate the impact of buyers' demands on the battery drainage of IoT devices under different scenarios through extensive simulations. Our results show that this approach is viable and benefits the provider and buyer by creating an autonomous and efficient marketplace for trading IoT data in real time from battery-powered IoT devices.

1.5.2 KYBChain: Know-your-buyer in Privacy-aware decentralized IoT data marketplace

In this contribution, we maintain three profiles for a buyer, namely practice, purchase and leakage, that capture his various characteristics regarding the provider's three concerns: non-compliance risk, data accumulation risk and leakage risk (**RQ4**). The practice profile is based on assessing the buyer's privacy and security measures and keeps track of any updates in his data protection practices. The purchase profile is a provider-dependent characteristic and monitors all the data purchase transactions of the buyer specific to a provider. A leakage profile maintains the records to assess the buyer's data leak risk based on the potential damage they can cause and the likelihood of its occurrence that depends on their past data leak events. However, even if providers have access to all these profiles, it will be difficult for them to digest the information within them. We implement a standardized and simple indicator for each buyer-provider pair, known as privacy rating (PR) (**RQ5**). This new privacy visualization metric measures the provider's overall and long-term risk caused by sharing his data with the buyer and limits subjectivity's role in the provider's decision. Our definition of privacy rating (PR) satisfies the following intuitive properties: better privacy practices and security measures reduce the provider's privacy risks and increase his PR; higher the sensitivity and visibility of accumulated data greater the provider's privacy risks and lower his PR; higher the probability and severity of data-leak risk, greater the provider's privacy risks and lower his PR. We identify the privacy attributes/elements to model these profiles and develop a methodology to formulate PR satisfying the above-stated properties. To address the lack of trust in buyer's information collation and its manipulation, we propose to use blockchain to record these profiles in an immutable, transparent, secured and decentralized manner (**RQ6**). We develop a blockchain-based data marketplace framework integrated with a privacy rating system known as KYBChain. KYBChain employs smart contracts to compute PR automatically, efficiently, accurately and timely. We conduct several experiments on synthetic data to demonstrate the efficacy and practical utility of PR. Furthermore, we present the proof of concept implementation of KYBChain in a private Ethereum network and analyze the overheads to demonstrate its feasibility (**IQ**). Our results justify the efficacy of privacy rating in aiding providers to make a privacy-aware decision about data sharing. Moreover, our evaluations of KYBChain reveal that the overheads introduced by our mechanism compared to a marketplace that does not incorporate a privacy rating system on the blockchain are insignificant relative to its privacy gains.

1.5.3 TrailChain: Traceability of Data Ownership across Multiple Data Marketplaces

In this contribution, we propose TrailChain which is a novel and fine-grained data ownership traceability mechanism by leveraging the power of blockchain technology on top of existing watermarking techniques. Our framework provides an immutable and trusted trade trail for tracking the sequence of data ownership spanning across multiple decentralized marketplaces in an automated, efficient and transparent manner. The novelty of TrailChain stems from (a) a mechanism that offers flexibility by identifying whether the resell/redistribution is legitimate/illegitimate or within or across marketplace systems (**RQ7**); (b) a fair resell payment sharing scheme that allows trusted, protected and automated sharing of resell revenue among the data owners in the trade trail over authorized reselling (**RQ8**); (c) a data ownership registration protocol that allows providers to register the ownership of original data and guarantees that the data has been generated by a genuine device. Besides, it also provides proof of data authenticity to the buyer by automatically verifying the trade lineage (**RQ9**); (d) a prototype implementation using four private Ethereum networks and simulation to demonstrate TrailChain's feasibility by benchmarking performance metrics including execution gas costs, execution time, latency and throughput (**IQ**). Qualitative security analysis of the architecture highlights its effectiveness in providing immunity to several common attacks. Finally, simulations demonstrate that our method detects reselling within the same marketplace and across different marketplaces. Besides, it identifies whether the reselling is authorized or unauthorized and fairly distributes the revenue among the data owners at marginal overhead.

1.6 Thesis Organizations

The thesis is organized as follows:

Chapter 2: This chapter presents the background and state of the art of related research.

Chapter 3: This chapter presents MartChain for enabling autonomy and efficiency in the decentralized IoT data marketplace while addressing the challenges of scalability, repudiation and the resource-constrained nature of IoT devices.

Chapter 4: This chapter presents KYBChain, which integrates a privacy rating system in a decentralized IoT data marketplace to record a buyer's practice, purchase and leakage profile and evaluate his privacy rating. Buyers' privacy rating empowers providers to act according to their preferences and make a privacy-aware informed decision about data sharing to them.

Chapter 5: This chapter presents TrailChain, a data ownership management framework that tracks data ownership within and across multiple marketplace systems and enables fair revenue sharing among actors registered in the same or different marketplaces.

Chapter 6: This chapter concludes this thesis and discusses directions for future research.

Chapter 2

Background

This chapter presents an overview of blockchain technology and related literature in the context of the application of blockchain in a decentralized IoT data marketplace. First, we present the overview of blockchain technology in section 2.1 and popular blockchain platforms in section 2.1.1. In the following sections, we focus on the state-of-the-art blockchain-based IoT data marketplace from autonomy, efficiency, privacy and traceability aspects. We considered industry-led solutions and academic-focused research frameworks and concluded each section with a comparative analysis of the discussed approaches. First, we discuss the autonomy and efficiency approaches in section 2.3.1. Second, we present the work on privacy in section 2.3.2. Third, the work related to the traceability of data ownership is discussed in section 2.3.3.

2.1 Blockchain Overview

Blockchain technology was introduced along with the first cryptocurrency, Bitcoin, by Satoshi Nakamoto. In his whitepaper [27], Nakamoto envisioned an electronic payment system using a distributed peer-to-peer network to enable direct transactions between users without needing a trusted third party. Bitcoin solves the problem of double-spending, i.e., spending the same digital token more than once and maintaining transaction order in a distributed system. Through its usage as the technical backbone for Bitcoin, blockchain has attracted tremendous attention from academia and industry in various other disciplines such as finance, supply chain, healthcare, law, IoT, and others. However, to utilise blockchain's full potential in these domains, it needs to be optimised according to its suitability in a particular industry. For optimisation, understanding the working of the blockchain is essential. In principle, a blockchain is an immutable digital ledger that publicly stores transactions (data records) after being verified by distributed nodes. These nodes produce timestamped transactions which get broadcast to the network, bundled into blocks and added to the append-only ledger after a mining process. The blocks form a linear sequence where each block references the previous block's hash, forming a chain of blocks. The ledger is replicated among the nodes, and all transactions are publicly visible, allowing network participants to verify the transactions' validity independently. Moreover, participants are pseudonymous, meaning they are identified by their account identifier, while their real identity remains anonymous. Thus, the blockchain structure enables features like distributed computation, transparency, anonymity, auditability, and immutability in a decentralized fashion. Next, we discuss the blockchain structure and its key components.

The basic structure of a blockchain is depicted in Fig. 2.1. Blockchain is a distributed data structure, similar to linked lists, that links blocks of data together, where the first block is called a genesis block. Each block is composed of a header and a body. The body consists of a set of transactions. A transaction represents an interaction between participants, and its structure consists of a unique ID, previous transaction ID, public key and single or multiple signatures. Based on the number of participants involved in generating the transactions, a valid transaction either includes the signature of one party, singlesig transactions or includes signatures of n out of m parties, multisig transactions. Most blockchain instantiations allow participants to create and use multiple public-privacy key pairs to preserve anonymity. The public key is used to verify the transaction, and the private key is used for signing the transaction. The transaction ID is the hash of its content. Each transaction is linked to the previous transaction made by the same

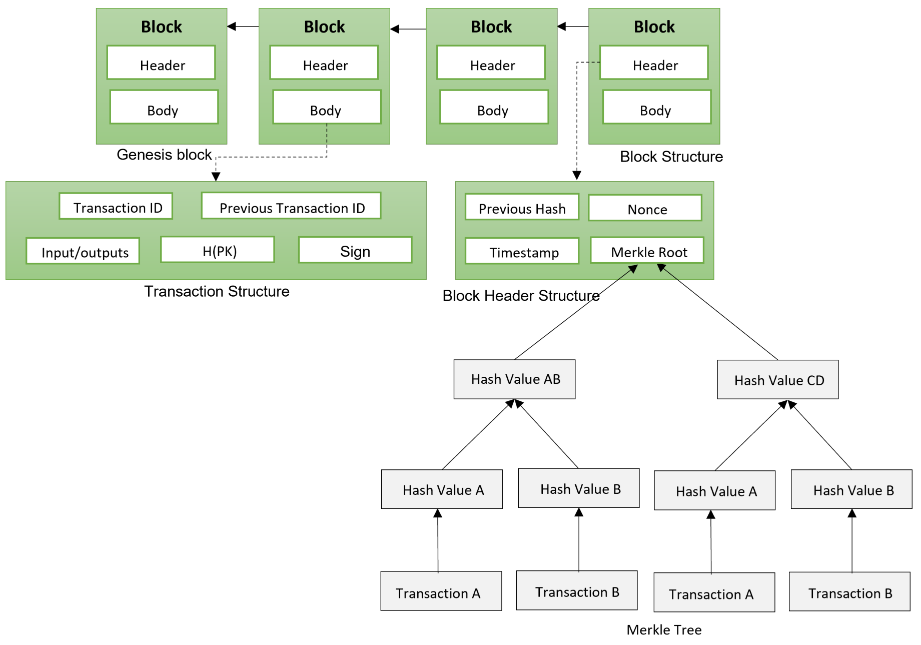


Figure 2.1: Structure of block and transaction

participant. The hash of the public key is stored to prevent the reconstruction of the private key from the party's public key. The block header contains the block number, the previous block header's hash value, block size, a timestamp, a nonce and the Merkle root. With the previous block's hash stored in the current block, blocks are cryptographically linked, and any tampering on the previous block will be detected. The timestamp is the time when a block is created. A nonce is used in the creation and verification of a block. The transactions within a block body are hashed in a Merkle tree. A Merkle tree is a binary tree with each leaf node labelled with the hash of one transaction and the non-leaf nodes labelled with the concatenation of the hash of its children. Merkle root is the root hash of a Merkle tree used to verify the transactions in a block efficiently. A small change in one transaction can produce a significantly different Merkle root. Hence, instead of verifying all the transactions, verification can be done by simply comparing the Merkle root.

Blockchain employs interconnected components organized into three layers to provide specific infrastructure features, as illustrated in Fig. 2.2. At the lowest layer, the blockchain has the signed transactions between participants. These transactions denote an agreement

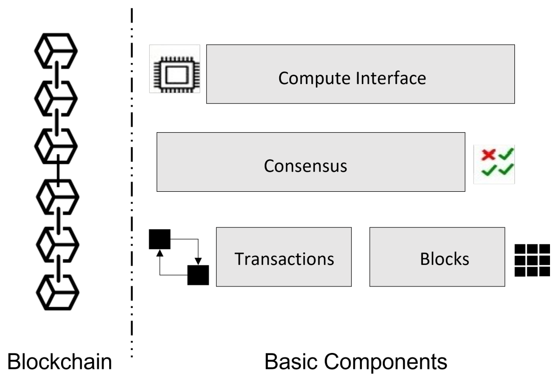


Figure 2.2: Components of blockchain infrastructure

between two participants, which may involve the transfer of physical or digital assets, completing a task, and other such tasks. These signed transactions disseminate to their neighbours. Any entity which connects to the blockchain is called a node. Nodes that fully validate transactions and blocks are called full nodes. These nodes verify the transactions according to rules, group the valid transactions into blocks, add them to the blockchain and receive a reward in return. Nodes must reach an agreement about the present state of the distributed ledger, which is the goal of the second consensus layer. Different consensus mechanisms exist to validate and add blocks depending on the blockchain type. These include Proof-of-work (PoW) [38] that requires nodes to expend resources solving a cryptographic puzzle. Proof-of-stake (PoS) [39] that randomly select validators according to how many coins they stake. Byzantine Fault Tolerance (BFT) [40] ensures that the distributed system remains fault-tolerant even in the presence of faulty or malicious nodes and others. The topmost layer, the compute interface, allows blockchain to offer more functionality. Typically, a blockchain stores the system's current state, and transactions trigger state transitions. In Bitcoin, the blockchain state consists of the user's balance. However, more advanced applications require storage of complex states, which are updated dynamically using distributed computing. The update is done by smart contracts, self-executing lines of code executed if certain conditions meet. The business logic or terms of the agreement are directly written into these lines of code.

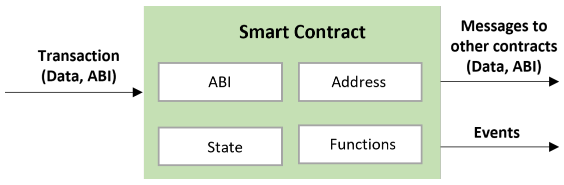


Figure 2.3: Structure of smart contract

Smart Contracts : A smart contract resides on a blockchain and is identified by a unique address. The structure of a smart contract is shown in Fig. 2.3. It includes a set of executable functions and state variables. These functions are executed by the transactions, which include input parameters required by the function. Upon the execution of a function, the state variables change depending on the logic implemented in the function. Smart contracts can be written in high-level languages (such as Solidity, Go, and Java). They are compiled into byte code using language-specific compilers and deployed on the blockchain with unique addresses that any blockchain user can call. Since a smart contract can consist of several functions, the application binary interface (ABI) must specify which function to invoke. Any user on the blockchain network can trigger the functions in the contract by sending transactions specifying the address and ABI to the contract. The contract code is executed on each node to verify new blocks.

2.1.1 Blockchain Platforms

In this section, we briefly describe different types of blockchain networks and outline the most popular and suitable platforms for IoT domains as identified in [41].

Blockchain networks can be categorized into three types: Public, Private and Consortium blockchain.

Public blockchain networks are permissionless in nature, which means anyone is free to join and participate in the blockchain network's core activities without needing permission from any authority. Any node within a permissionless blockchain network can read and write to the ledger. Since such networks are open for anyone to join, malicious nodes may attempt to publish blocks that subvert the system. To prevent this, permissionless blockchain networks often utilize a consensus mechanism such as POW, POS etc., that requires nodes to expend or maintain resources when attempting to publish blocks. Popular public blockchain includes Bitcoin, Ethereum, Litecoin, IOTA etc.

Private blockchain networks are permission-enabled blockchain networks managed by a single organization/participant with complete control over the blockchain's rules. In a private blockchain, the central authority determines who can join the network. The central authority does not necessarily grant each node equal rights to perform functions. Private blockchain networks are only partially decentralized because public access to these networks is restricted. A common example of a private blockchain is Multichain.

Consortium blockchain networks are permission-enabled blockchain networks governed by a group of organizations rather than one entity. Since only authorized users maintain the blockchain, there is a certain level of trust in each other. It is also possible to restrict read access and who can issue transactions. They use consensus mechanisms, such as BFT, for publishing blocks which often do not require the expense of resources. Unlike in public blockchain, the identity of every member is known in consortium blockchain, hence, misbehaving members' authorization can be revoked. Examples of consortium blockchains include Hyperledger Fabric, Corda, Quorum etc.

Next, we briefly outline some popular blockchain platforms used for application development in various industries. We also outline the specific platform we used to implement and evaluate the ideas proposed in this thesis.

- **Ethereum** [42] is a public distributed computing platform that enables developers to build and deploy smart contracts to develop decentralized applications. Ethereum provides an underlying technology, Ethereum Virtual Machine or EVM, a runtime environment for compiling and deploying smart contracts. The Ethereum blockchain is powered by ether, its native cryptocurrency. Ethereum faces a scalability problem resulting in high transaction fees and low transactions per second of 15 tps. Moreover, it uses POW, which typically requires a high amount of energy. Recently, Ethereum 2.0 was launched based on POS and is much more efficient, with a capacity to handle up to 100,000 tps. Ethereum 2.0 also provides sharding that addresses the scalability concerns associated with classic Ethereum. Ethereum is most suitable for products and applications running on an open network where participating entities are not subject to access control. Some examples are decentralized finance, digital identity, payments, and others.

- **IOTA** [43] is a cryptocurrency based on a directed acyclic graph structure, the Tangle, which has the potential for scalability and zero transaction fees. On top of the Tangle network, Masked Authenticated Messaging (MAM) [44] allows transmission, access, and verification of an encrypted data stream. Based on Tangle and MAM, the IOTA Foundation launched a data marketplace [45] prototype that allows storing of data on Tangle and enables trading where privacy and integrity meet with MAM.

- **Hyperledger Fabric** [46] is an open-source project built by IBM and Linux Foundation that aims to offer a modular and extendable framework. Its distinguishing modular feature allows a pluggable architecture that includes components like a consensus, membership services, endorsement policies, etc. The consensus of the earlier version of Fabric was based on the order-execute model, Kafka, wherein Fabric executes the transactions before finally committing them. Hyperledger Fabric v2.0 introduced Raft, which follows a leader-follower model wherein a leader node is elected for every channel, and the follower nodes replicate the leader's decision. Raft is better than Kafka in terms of transactions per second due to its simpler framework than Kafka. Hyperledger Fabric uses Chaincode as the technology's business logic, which uses FabToken as a native token. The fabric has vast applications across industries, from healthcare to supply chain to real estate to banking.

- **Corda** [47] is an open-source R3 consortium product that is developed to record and automate legal agreements in financial and banking environments. It uses the concept of state changes and transactions instead of blocks and chains. It aims to achieve scalability and address the privacy concerns faced by banks or financial institutions. To do so, Corda employs Notaries, either centralized or distributed entities, who validate the transactions. Corda's smart contracts support code and legal prose known as the Ricardian Contract. Ricardian Contracts function as legally binding agreements between two parties based on agreed-upon terms and conditions. A contract is cryptographically signed and verified using the blockchain and readable by lawyers and machines. Corda offers a Token SDK that enables the creation of native tokens on the blockchain. Though Corda was initially designed for the financial service industry, it is gaining popularity in the construction industry, healthcare, energy sector, government technology and many others.

- **Quorum** [48] is soft forked from Ethereum, developed by JP Morgan. It aims to provide the finance industry with a permission-enabled enterprise blockchain that supports transaction and contract privacy. Quorum uses a consensus protocol called "QuorumChain", wherein a consensus is reached by simple majority voting. Quorum uses Raft-based and BFT for better fault tolerance, faster block time and better transaction finality. Quorum supports both public and private contracts. Anyone on the network can execute the public contracts, while only specific nodes have access to execute the private contracts. It supports JPM Coin as the native cryptocurrency. Some of Quorum blockchain's most prominent use cases are banking and finance, insurance, enterprise solutions, travel and hospitality, and many others.

- **Multichain** [49] is a private blockchain extended and forked from Bitcoin. It consists of two types of nodes with different capabilities. Admin nodes are responsible for creating filters, issuing assets, and approving new filters and new nodes. Member can send and receive transactions. Like Bitcoin, MultiChain uses POW consensus but enforces a round-robin schedule to promote mining diversity. It allows configurable target time for adding a block, i.e., less than 10 minutes which is an average time for Bitcoin. MultiChain enables an institution to create multiple private blockchains on the same machines, allowing nodes to pass data between different blockchains on the same network. Unlike other blockchain platforms, MultiChain supports multiple tokens. It also supports a smart filter, a piece of code that allows custom rules to be defined regarding the validity of transactions or data. MultiChain is used to develop lightweight financial systems, provenance tracking in the supply chain, inter-organizational record keeping, and many others.

- **Litecoin** [50] is an open-source blockchain, forked from Bitcoin. Technically, Litecoin is identical to Bitcoin but features faster transaction confirmation times and improved storage efficiency due to the reduction of the block time (from 10 minutes to 2.5mins) and the use of Scrypt PoW, a less resource-intensive consensus method. Litecoin adopts a new feature of segregated witness to limit the number of orphaned blocks, which are formed when two miners mine a block at nearly the same time and then reject one. It enables interoperability and allows trading multiple cryptocurrencies using atomic swaps. Litecoin also implemented a lightning network providing a valuable second layer (Layer 2) of transactions to process 840,000 tps compared to 7 tps in Bitcoin. Litecoin is a digital currency used for instant payment.

In this thesis, we chose Ethereum for implementing and evaluating all proposed ideas (Chapters 3, 4, 5) due to its flexibility to build both public or private blockchain networks, ease of deployment, and availability of tools for blockchain deployment, management, data query, smart contract implementation etc. However, our proposed ideas are blockchain-agnostic and can be implemented in any blockchain instantiation that supports smart contract execution, for instance, Corda, Quorum and Hyperledger Fabric.

2.2 IoT Data Marketplace

Recall from Chapter 1, that an IoT data marketplace: (i) addresses the issue of data silos and enables sellers to monetize and distribute the data generated from their IoT devices; (ii) addresses the issue of discovering and accessing new data sources and enables buyers to navigate through data offerings and find the one which is needed. In general, a data marketplace acts as a platform to facilitate data trade by providing various functionalities. To be functional as a marketplace, authors in [51] identify components of a marketplace, depicted in Fig. 2.4, which are described as follows:

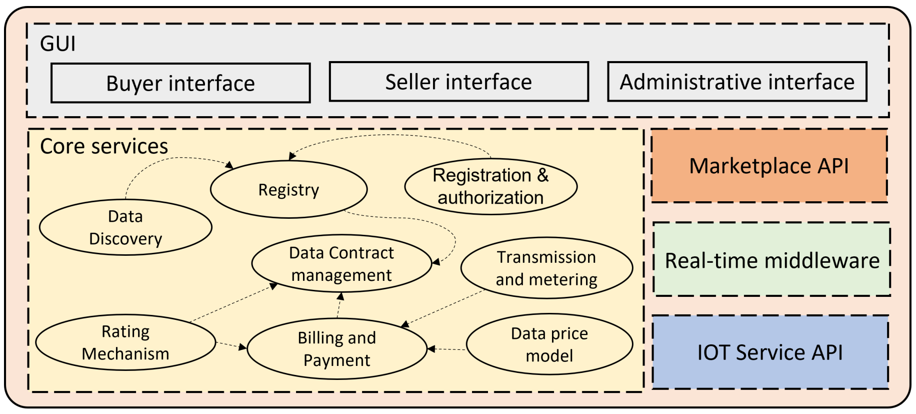


Figure 2.4: Components of data marketplace

1. *User registration and authorization:* This allows users to register in the marketplace either as sellers or buyers. A seller owns data, while a buyer is interested in purchasing a particular data type. The seller may implement different access granularities and only allow specific buyers to access data based on his usage policy.

2. *Data discovery and selection:* This allows buyers to search and select sellers' devices based on their desired data type (e.g., traffic data or solar panel data).

3. *Data price model:* This allows the seller to select a pricing strategy with typical options being fixed, tiered, dynamic, or negotiated pricing. In a fixed pricing model, the price is fixed for a particular duration, while in tiered pricing, the price is fixed but includes multiple categories. Dynamic pricing changes the price over time, while in negotiated pricing, the price is actively negotiated at the time of sale.

4. *Contract management:* The seller and buyer employ this to manage the trade agreement. A contract usually consists of terms and conditions that govern the quality of data and the associated costs.

5. *Rating mechanism:* The buyers and sellers employ this function to rate each other, which establishes a level of trust.

6. *Data metering:* This is used to measure particular parameters (e.g., amount, frequency, and type) associated with an ongoing trade for billing purposes.

7. *Billing and payments:* This manages the payment process for trades. The marketplace may incorporate different mechanisms of payment (e.g., advance payment, pay-as-you-go, and payment at predetermined intervals).

8. *Real-time middleware:* This incorporates different functions related to real-time data streaming, such as data routing, resource management, and encryption.

Data marketplaces can further be distinguished based on a centralised or decentralised architecture. Conventional data marketplaces [21, 24, 52] rely on centralised architecture where a central trusted third party (TTP), i.e., a particular data marketplace operator, oversees the trade to ensure that both parties commit to their obligations, which results in several limitations such as scalability, limited privacy, a central point of failure, and low security (see Chapter 1). A decentralised IoT data marketplace allows participants to trade data without having to trust a third party. Next, we discuss the proposals for realising decentralised IoT data marketplaces using blockchain technology.

2.3 Blockchain to enable Autonomy, Efficiency, Privacy and Traceability

In the following sections, we explore the application of blockchain technology for providing autonomy, efficiency, privacy and traceability in a decentralised IoT data marketplace.

2.3.1 Autonomy and efficiency

This section explores the value proposition of blockchain technology for autonomy and efficiency in a decentralized data marketplace. In most of the state-of-the-art, blockchain offers certain functionalities in the broader context of data marketplaces. For example, blockchain is used as a trade transaction management system in [53] to improve transparency and traceability. Making use of the immutability and distributed nature in [11], providers use blockchain as a product catalogue that buyers can browse to find the proper data type based on their requirements. In another approach [54], blockchain smart contract is used to implement an access control mechanism enabling the providers to manage who can access their data and for how long. In the following section, we group existing solutions based on their design considerations in architecting a blockchain solution for enabling autonomy and efficiency in marketplace functionalities. These approaches include trade transaction management, distributed data catalogue, hybrid centralized-decentralized architectures, decentralized data storage and access mechanisms, and agreement instantiations. We highlight the core features of each approach and discuss some relevant works.

Trade transaction management : The purpose of a marketplace is to facilitate the trade of a particular item between users. We define a trade transaction as an exchange of value between a buyer and seller that involves the user's communication, negotiation, execution, settlement, and payment. In the trade management framework, the entire trade transaction history is recorded in the blockchain, which enables the participants to trace the history of the transactions based on the transaction time stamp and have complete visibility of the communications and data exchanges, leading to a high level of transparency. Benefiting from the inherent immutability offered by blockchain, when a transaction is confirmed, either party (i.e., seller or buyer) cannot modify the content or data, which provides a trusted trading platform.

In [29], a secure publish/subscribe (SPS) service protects the users' anonymity and achieves fair payment in exchanging CPS data. All the interactions in SPS between publishers and subscribers, from publishing a topic to specifying interest, payment and rating, are recorded in the blockchain. In [53], a brokered-based IoT data trading employed smart contracts to achieve non-repudiability and transparency by recording the following information in the blockchain: participant details, provider data offerings, trade agreements to define the terms of the data exchange and data receipt during the data delivery. In another approach [30], SDN controllers provide storage servers where the seller uploads its encrypted data, and the blockchain module employs smart contracts to automate data monetization by using several transactions: advertisement transaction to record the provider's data price and description; purchase transaction to record the buyer's interest and amount paid; and response transaction containing a symmetric key encrypted with the buyer's public key. In [55], a four-layered data exchange framework was proposed: the data layer comprises IoT data stored in the cloud, the network layer comprises nodes, storage, servers etc., connected in a P2P fashion, the interaction layer includes web and mobile interfaces for participants interaction, and the management layer leverages smart contracts to manage and record users, data collection and search, data access rights and transaction history in the blockchain. In [37], providers store their IoT data in an encrypted form in Swarm, a decentralized storage system. The smart contract provided various ABIs to register users, add/query devices, purchase data and define data structures to record payload and user information on the blockchain. In other work, Block-DM [32] is based on the cloud for storing data, and a supervising authority (SA) maintains the consortium blockchain. SA assigns anonymous credentials to providers, enforces providers' consent-based data marketing over IoT data and records all operations in the blockchain.

Blockchain-based marketplaces facilitate the trade management process, which involves storing the transaction's history in a transparent and immutable manner. In a data marketplace, it is critical to maintain a catalogue of the available data types and match them properly with the potential buyers, outlined in the next section.

Distributed Data Catalogues : In a blockchain-based marketplace, the participants in the trade are connected and share information through blockchain, which eliminates the need for central controllers. The sellers advertise features of their data, including price, data types, and reviews in the blockchain. The buyers can explore/browse the chain to find the proper data type and connect to the seller to conduct the trade. Blockchain enhances the reliability of the listing process as data offerings are replicated in all the participants in an immutable manner. As individual sellers manage the listings, the likelihood of errors is low, given that malicious entities in the marketplace cannot alter the price or the preference of any listing to benefit other sellers.

The blockchain in [11] functions as a product catalogue that buyers can search to find suitable data products. Each seller creates a detailed product description, including data type, seller's id, price, and IP address and stores it in a distributed file storage (DFS) framework such as IPFS or Storj. The DFS returns a storage identifier that the seller then stores along with product metadata in the blockchain through a *product smart contract*. A similar approach is proposed in [36] where a seller uploads the product image and description to the IPFS and then stores the product information and IPFS links onto the blockchain, and buyers place their bid to buy data. In another work [56], a smart contract is used to realize data offering listing and discovery functionalities. Provider issues add transactions to store his data offerings on the chain, and buyers issues search requests to find the offering. *Offering smart contract* provides two query functions that return either the ID of the first offering that matches the criteria or a list of IDs of all the offerings matching the criteria. The search happens off-chain, where the node keeps track of which blocks contain which offerings. In other work, data is stored in the IOTA blockchain as message streams using the MAM protocol. Every single message is indexed by a keyword set to discover the content of specific data/MAM channels available in the Tangle. Hypercube-based Distributed Hash Table (DHT) lookup functionalities are used to search data through keywords on MAM channels. Several industry projects based on the distributed data catalogues approach exist to support IoT data exchange. Databroker [57] relies on the gateway operator that runs the Ethereum node and DataBroker API to expose the gateway allowing sensor owners to sell their data on the platform. Sensor owners upload their data offerings on the marketplace, and buyers can discover and buy access to these data by paying with DXT tokens. Another similar platform for sensor data delivery is Moeco [58], which acts as a gateway for the providers to sell their data to the buyers acting as a middleman and processing the payment on their behalf. Streamr [59] uses a time-based brokered subscription model and DATA token to trade real-time data streams. Using *marketplace smart contract*, providers store a registry of datasets and coordinate access and permissions.

The listing services allow the providers to advertise their data and the buyers to search and retrieve particular data types. However, as outlined in section 2.2, the data marketplace involves design decisions beyond data-listing services. The following section discusses marketplace architectures.

Hybrid Centralized-Decentralized Architectures : The existing data marketplace solutions can be categorised as centralised, where a central controller manages all aspects of the trade and decentralised, where the core functionalities of the marketplace are distributed between the participants. As outlined in Chapter 1, the central solutions introduce privacy, security, and single-point-of-failure challenges. A purely distributed method (i.e., blockchain-based method) also becomes challenging as storing all the indexing and management information in all the participants increases the storage and thus increases the cost of managing the chain. A blockchain-agnostic method using a hybrid approach (i.e., less critical centralised and new decentralised elements) creates a practical and realistic solution for data marketplaces. Hybrid approaches can either rely on brokers to manage trades or rely on the central server to manage to trade.

The data monetization framework in [54] uses the Ethereum blockchain to enable providers to create an IoT device contract to facilitate payment settlement, data price and usage time auditing, issuance of access tokens, and events log. Besides, the framework also employs the message queuing telemetry transport (MQTT) broker, a TTP hosted in the cloud, to improve the reliability, scalability, availability, and accessibility of the framework with low latency. MQTT broker aggregates the IoT data, authenticates the buyer off-chain based on the unique token issued by the IoT device contracts and gives data access to him. Similar to the above approach, [60] proposed a review system based on blockchain to record metadata, reviews and ratings. The MQTT broker stores the encrypted data and facilitates data transfer between providers and buyers. Another work [61] uses an HTTP server-based platform, Hermes, which employs IOTA to store the provider's data. The marketplace server facilitates trading by providing APIs to register or de-register users; post queries or offers; forward decryption keys of the stream on IOTA, encrypted with buyers' public keys, to buyers; and resolve disputes. In other work [33], the medical data-sharing framework relies on a centralized marketplace to provide matchmaking services and receives a commission in return. A consortium blockchain, maintained by academicians, manages data access requests, ensures data integrity, verifies the provider's data-sharing policy, and payment settlement mediated by the marketplace. A similar approach in [62] uses a mediator to facilitate trade by matching buyers with sellers in return for a percentage of incentive on successful matchmaking. Blockchain record all the agreements, settlements, and penalties for misbehaviour. In [31], the authors leveraged blockchain for IoT traffic metering and contract compliance on top of the IoT-brokered data infrastructure. The broker mediates all the interactions between publisher and consumer, manages all the subscriptions and reliably delivers messages to the subscriber. An industry project, Wibson [63], relies on Notaries to authenticate and validate the data made available by the seller and resolve conflicts between buyer and seller. The sellers install a smartphone application and can sell their data in exchange for a Wibson token.

The hybrid method is a combination of centralised and decentralised models that facilitates the functionality of the data marketplace by reducing overheads and increasing trust and security. In the next section, we discuss decentralised data storage and managing access to the data, which is critical to a decentralised marketplace.

Decentralized Data Storage and Access Mechanisms : An access control mechanism enables the data owners to manage who can access their data and for how long. The access mechanism separates the data exchange into two parts: raw data exchange, where the participants share the raw data, and right access exchange, where permission to access particular data is exchanged between participants.

Sash [64] employs smart contracts to evaluate access requests of buyers based on the provider's predefined policies. Providers store their data in the cloud encrypted using prefix encryption and advertise data offerings to the blockchain. A key distribution authority uses the Access Control List (ACL) recorded in the blockchain to decide whether or not to share the master secret key with the buyer. A similar work [35], PrivacyGuard, leverages smart contracts to enable providers' data access control. It executes the smart contract off-chain in a trusted execution environment based on Intel SGX to prevent exposing user data on the public blockchain. In [65], a secure and auditable data management consists of a distributed hash table (DHT) to store chunks of IoT streams as a key-value pair, and blockchain empowers data owners by managing access rights per data stream basis for a limited time. In [34], providers store their data in the interplanetary file system (IPFS) and receive a hash in return. They divide the hash into n-shares and record the encrypted hash shares on the blockchain. A passive entity, the worker, authenticates the buyer's identity, queries the smart contract and sends the decrypted hash shares to the buyer. An industrial project, datapace [66] is a marketplace for IoT sensor data based on Hyperledger Fabric to ensure data integrity using policy-based data verification. It uses Mainflux [67] as an IoT cloud platform enabling connectivity and management of IoT devices via the gateway. It leverages smart contracts to define the conditions under which the data is traded in exchange for a TAS token. Another project, Datum [68], relies on DAT Token Smart Contract to provide secure data trading. Providers submit their data by connecting different IoT services to the platform, paying a small DAT token fee to store it in the network. Buyers acquire data for DAT tokens, and the smart contract initiates an off-chain key exchange under the provider's terms. Data is encrypted under the provider's usage terms, anonymized and stored on the storage node miners that earn DAT tokens for saving and transmitting the encrypted data.

Access control management is critical to ensure that the user controls the stored data and thus protects their privacy. The seller and buyer must reach an agreement to access data, discussed in the next section.

Agreement Instantiation : An agreement instantiation approach manages the agreement life cycle and ensures the agreement's integrity, non-repudiation of the participants, and authentication.

An agreement framework [69] that relies on a data marketplace used as a TTP handling all transactions among all stakeholders, including providers, data custodians, and buyers.

Blockchain is to record the agreements established in the data marketplace and monitor the activities within the ecosystem. Data custodians collect data from the provider and transfer data assets to the buyer. The provider broadcasts an agreement in the blockchain that describes their acceptable terms and conditions. The data custodians broadcast a service-level agreement based on negotiated terms with the provider and asset terms and conditions. The marketplace creates a transaction that contains the agreements. The involved parties validate the transactions and store the corresponding hash in the blockchain. The data and service assets are then saved in the service repository with the link to their respective agreement's hash.

The summary of the literature discussed in this section is provided in Table 2.1. We compare the existing work along these dimensions: approach, system model, platform, smart contract usage, autonomy level (i.e., what level of autonomy the marketplace provides to its participants), IoT efficiency for trading data in real-time and scalability. We can note from Table 2.1 that the existing solutions do not cover all the essential dimensions. Moreover, IoT efficiency is not considered necessary while trading data stream in real-time from the resource-constrained IoT devices in the existing IoT data marketplace systems. With a trade-off with scalability, some frameworks provide full autonomy for participants.

We address the challenges mentioned above by proposing a MartChain framework (see Chapter 3). MartChain aims to provide full autonomy in data trading using a public blockchain network. It also addresses the issue of scalability by employing multiple trustless geographically distributed facilitators. Moreover, MartChain is underpinned with an optimization module to achieve energy efficiency for resource constraints IoT devices for trading data stream in real-time.

Table 2.1: Comparison of autonomy and efficiency approaches in a decentralized IoT data marketplace

Article	Approach	System model	Platform	Smart contract usage	Autonomy	Efficiency	Scalability
[29]	Trade transaction	Publish-subscribe model	Bitcoin	Setup, publish, subscribe, match, payment, reputation	Full	×	×
[53]	Trade transaction	Broker for data transfer	Ethereum	Data offering, trade agreement, data receipt, settlement	Full	×	×
[30]	Trade transaction	SDN for data storage and transfer	Hyperledger	Data advertisement, purchase request, data access, settlement	Full	×	×
[55]	Trade transaction	Four layer: data, network, management, interaction	Ethereum	User management, data management, exchange management	Full	×	×
[32]	Trade transactions	Cloud servers to store data	Hyperledger	Data listings, data trading interactions	Full	×	×
[11]	Data catalogue	DFS for storing data, SDPP [70] for payment transfer	Ethereum	Decentralized registry for data offerings	×	×	×
[36]	Data catalogue	Bidding system based on IPFS as data storage	Ethereum	Product information, user bids, escrow account	×	×	×
[37]	Data catalogue	Swarm as data storage, Raiden micropayment as payment channel	Ethereum	Authorization mechanism, data offerings	×	×	×
[56]	Data catalogue	×Nodes maintains local copy of offerings for off-chain search	Ethereum	Data offerings and retrieval	×	×	×
[71]	Data catalogue	DHT for keyword-based search	IOTA	Data offerings	Partial	×	×
[54]	Hybrid model	MQTT broker for data transfer	Ethereum	Access token, data offerings	Partial	✓	×
[61]	Hybrid model	Server manages user registration, data offerings, queries, disputes	IOTA	Payment settlement	×	✓	×
[60]	Hybrid model	MQTT broker stores and transfer data, trusted arbitrator for disputes	Ethereum	Data metadata, reviews and ratings	×	✓	×
[31]	Hybrid	Publish-subscribe model, broker mediate interactions	Ethereum	Payment settlement	×	✓	✓
[62]	Hybrid model	Mediator mediates all interactions	N/A	Agreements, settlement, penalizing for misbehaviour	Partial	✓	✓
[33]	Access control	Cloud for storing data, Broker for facilitating marketplace functionality	N/A	user registration, data sharing policy, payment, recording all transactions	Partial	×	✓
[64]	Access control	Cloud storage to store data, key authority to share secret keys for decryption	Hyperledger	Add and verify metadata containing ACL	Partial	×	×
[65]	Access control	DHT for data store and transfer	N/A	Access control	Partial	×	×
[34]	Access control	IPFS for storing data, workers mediate all interaction with a smart contract on behalf of customer	Ethereum	Recording encrypted hash shares, access control, payment, review	Partial	×	×
[35]	Access control	TEE based on Intel SGX to execute smart contract off-chain, data broker to address scalability	Ethereum	Data access policy	Partial	×	✓
[69]	Agreement instantiation	Server for marketplace services, data custodian to store and transfer data	N/A	Agreement, transaction logs	Partial	×	×

2.3.2 Privacy

This section discusses the state-of-the-art for enabling privacy in decentralized IoT data marketplaces. We first discuss the existing solutions that devised proposals using privacy-enhancing technologies (PETs). In the following sections, we present related work of privacy awareness in the IoT ecosystem and label, rating, and scoring system.

As discussed in Chapter 1, exchanging IoT data in the marketplace entails the risk of exposing individuals' sensitive information that may restrain them from sharing data. Many initiatives explored privacy-enhancing technologies to enhance user's privacy while preserving data utility. These includes digital signatures, differential privacy, anonymization techniques such as k-anonymity, encryption, trusted execution environments, zero-knowledge proofs, and hashing. Digital signatures ensure data authenticity and integrity. Differential privacy adds noise to data to protect individual identities. Anonymization techniques such as k-anonymity masks individual identities by grouping individuals with similar attributes. Encryption protects data in transit and at rest. Trusted execution environments ensure secure data processing. Zero-knowledge proofs enable data validation without revealing the data, and hashing securely stores and verifies data. Several existing works based on these privacy-enhancing technologies are discussed below.

In [72], the authors use several cryptographic primitives to realize practical security. These include symmetric encryption to encrypt the provider's data and asymmetric encryption for digital signatures to ensure that the data source is authentic as claimed. The hashing mechanism verifies the integrity of transferred data by making the hash public before transferring the data. In [73], cloud servers store the provider's encrypted sensor data, establish runtime dynamic smart contracts between provider and buyer and use a proxy re-encryption scheme to transfer the anonymous data securely to the buyer. In other work [74], Sterling enables buyers to perform privacy-preserving analytics and apply machine learning (ML) over private data through smart contracts, trusted execution environments and differential privacy. In [75], data price is evaluated using differential privacy based on the provider's privacy loss in trading data. A provider deploys a compensation contract in the Quorum platform that defines the function of data and privacy loss and also sets the maximum privacy budgets for a dataset. The compensation contract manages the privacy budget and allocates it according to the buyer's accuracy requirements. Another work [76] used two data protection techniques to provide k-anonymity and differential privacy properties in a Sensing-as-a-Service model to ensure anonymity. Through a specific smart contract where parties agree on predefined policies, an aggregator receives access to data from k data subjects. The aggregator provides the sensing service and produces an anonymized dataset that buyers can acquire. In [77], the protocol integrates ring signatures to enhance the privacy of provider identity, extends the double authentication preventing signature (DAPS) for fair data trading and utilizes similarity learning to guarantee the availability of trading data. SDTE [78] consists of SDTP, a secure blockchain for contract deployment, requirements matching and secure contract execution on Intel's Software Guard Extensions (SGX). SDTP consists of trusted nodes (SGX supported) and normal nodes (non-SGX). The buyer deploys his data analysis contracts on SDTP, finds some sellers based on the data of interest, and finds SDTP's trusted nodes. Then, the seller sends the data to the trusted nodes selected by the buyer. Data analysis will be performed on the trusted nodes, and the execution results will be sent to the buyer. An industry project, Agora [79] leverages cryptographic techniques such as FE scheme to achieve data privacy, tailored Zero-knowledge proof (ZKP) ensures output verifiability, and atomicity of payments by leveraging smart contracts.

Although privacy-enhancing technologies have received tremendous attention, they are not without their downsides as presented in [80]. Firstly, they can be complex and resource-intensive, which can make them difficult and expensive to implement. Additionally, PETs can sometimes be prone to errors, which can increase the risk of privacy breaches. Furthermore, some PETs may not be fully effective at protecting privacy, as attackers may still be able to use techniques such as inference attacks to gain insights into sensitive information. Finally, PETs can also limit the utility of data, as they may require certain data to be obfuscated or encrypted, which can make it difficult to extract valuable insights. Moreover, in the IoT data marketplace, providers often share their IoT data with third-party buyers who may use the data for purposes beyond the provider's expectations. For example, using advanced technologies like data mining, AI, and ML, a malicious buyer can make unauthorized predictions about a provider's behaviour. Moreover, depending on how often and to what extent buyer collects the provider's data, the privacy threats including identity theft, financial loss, or reputation damage increases. These privacy harms and risks are more for providers in the IoT data marketplace since buyers may track habits, behaviours, and locations through the excessive and ubiquitous data collection. Therefore, to address the above issues, there is a need for increased privacy risk awareness in the data marketplace. By informing providers about the potential privacy risks associated with sharing their IoT data, they can make informed decisions based on their privacy preferences and needs. This can help to reduce the likelihood of privacy harms and increase user trust in the marketplace. Additionally, this can help to promote responsible data sharing practices that are consistent with legal and ethical standards. Overall, making privacy risk awareness crucial for protecting users' privacy and promoting a trustworthy IoT data marketplace.

Privacy-awareness in IoT ecosystem : There are several proposals for privacy risk assessment in the IoT ecosystem [81–84] to inform users about privacy risks associated with the disclosure of their personal information to third parties. In [85], the authors discuss the key requirements for the design of privacy risk-aware frameworks around wearable and IoT ecosystems. Their framework enables the users to make informed data-sharing decisions by evaluating privacy in terms of risk-benefit trade-offs. Several other works [86,87] enable users to define privacy-aware policies for data sharing based on their preferences. In [88], authors propose a policy framework for user data sharing based on the user-editable and negotiable privacy policy that defines the purpose, type of data, retention period and price. Other works [89, 90] developed a decision model for users to guide their choices regarding data sharing. Current data marketplace ecosystems rely on trust management models [33, 60, 91] where the provider's decision whether to give data access to the buyer depends on the buyer's trust score. However, trust is a subjective belief about an entity in a particular context [92]. It lacks factual and relevant information about buyers that can empower providers to properly assess privacy risk and make a privacy-aware informed decision about data sharing. To design such a mechanism for providers in the data marketplace, understanding their preferences and concerns is of great importance. Factors such as trade interactions, transparency of mechanisms, context and who is collecting the data are essential determinants that are not considered in existing approaches.

Label, rating, and scoring system : Labels, ratings and scores are a common approach in contexts such as food [93] and energy [94] to communicate important information effectively, have been shown to impact users' purchase decisions significantly. In the privacy context, Apple [95] and Google [96] have recently required application developers to provide information about the privacy practices of their applications. This information is presented on a private label in the app and play store to help users make a more informed application selection. Various studies [97–99] have shown that privacy and security labels for IoT devices could effectively inform users about the privacy risk associated with data collection. Authors in [100] propose a privacy and security label for the internet safety of IoT toys to help parents make purchase decisions. Other studies [101, 102] compute the privacy score of a user indicating his potential risk caused by his daily information-sharing activities on social networking sites. An automated website scanning portal, PrivacyScore [103], develops a benchmark tool to assess website security and privacy features. The framework introduced in [104] produces a Privacy Invasion Profile for each mobile application by analyzing its permissions, which can be used to compare the risk of privacy invasion for a specific application. However, these rating systems are based on a centralized model where data to compute ratings are stored on the central server. The rating data can be modified and manipulated to change the rating. Many studies employ blockchain to develop rating systems in various application domains to address this issue. These include rating restaurants on website [105], assessing the success level of tourism destinations [106], movies rating for deciding watch [107], credit rating for assessing the creditworthiness of a borrower [108–110]. A rating system proposed in [111] rate service providers to protect IoT devices from unreliable services by them.

Table 2.2 summarizes the existing approaches to address the privacy issue in data sharing. We compared the discussed articles against the following categories: use case, architecture, adopted privacy mechanism, allow a provider to make an informed decision for data sharing and if they use any rating/label/scores. It is evident from the table that the existing solutions to address privacy challenges in a decentralized IoT data marketplace are based on privacy-enhancing technologies that allow providers to protect the privacy of their IoT data. Other solutions employ privacy and security awareness mechanisms to allow users to make an informed decision about data sharing by assessing their privacy risk and acting as per their preference using labels/scores/ratings. However, these frameworks are centralized, making the risk assessment or rating data prone to manipulation. Furthermore, these approaches are based on data sharing in different contexts, such as mobile applications, social networks, wearable IoT devices, and websites. They do not consider factors suitable to assess provider's risks associated with sharing their data and

Table 2.2: Comparison of privacy approaches in a decentralized IoT data marketplace

Article	**Year**	**Use-case**	**Architecture**	**Privacy mechanism**	**Informed decision**	**Score/Label/ Rating**
[72]	2020	Data marketplace	Decentralized	Digital signature, asymmetric encryption, hashing	×	×
[73]	2021	Data marketplace	Decentralized	Proxy re-encryption scheme	×	×
[74]	2018	Data marketplace	Decentralized	Trusted execution environments and differential privacy	×	×
[75]	2020	Data marketplace	Decentralized	Differential privacy	×	×
[76]	2020	Data marketplace	Decentralized	k-anonymity and differential privacy	×	×
[77]	2019	Data marketplace	Decentralized	Ring signatures, double authentication preventing signature	×	×
[78]	2019	Data marketplace	Decentralized	Trusted execution environment	×	×
[112]	2017	Data marketplace	Decentralized	k-anonymity, hashing	×	×
[79]	2021	Data marketplace	Decentralized	Functional encryption scheme, Zero-knowledge proof	×	×
[85]	2020	Wearable IoT devices	Centralized	Privacy risk-aware	✓	×
[86]	2020	IoT data sharing	Centralized	Privacy-aware policies	✓	✓
[100]	2021	IoT toys	Centralized	Privacy and security aware	✓	✓
[101, 102]	2010, 2017	Social networking sites	Centralized	Privacy risk-aware	✓	✓
[103]	2017	Website	Centralized	Privacy and security aware	✓	✓
[104]	2014	Mobile application	Centralized	Privacy risk-aware	✓	✓
[111]	2019	IoT SPs credibility	Decentralized	N/A	✓	✓

make a decision based on their preferences and needs in the data marketplace.

Our proposed privacy-risk-aware solution KYBchain (see Chapter 4) addresses the above-mentioned challenges. KYBchain approaches the privacy issue from the provider's viewpoint and empowers them to make privacy-aware data-sharing decisions. It computes the privacy rating of the buyer that the providers can use to manage their long-term privacy risks associated with trading or sharing their IoT data in the marketplace. In the proposed solutions, extensive performance evaluations are carried out to demonstrate the efficacy of privacy ratings. We also presented proof of concept in a private Ethereum network and demonstrated the feasibility of KYBchain using Hyperledger Caliper. In addition, we have also made use of some PETs, including digital signatures, encryption techniques, and hashing, to protect the provider's data and minimize privacy risks. However, in the current research, we have not considered incorporating ZKP-based solutions because the privacy provided by them comes at the cost of increased computation and communication overhead. In the future, we plan to explore ZKP-based solutions in the proposed architecture and investigate the trade-off between privacy and efficiency.

2.3.3 Traceability

Blockchain can play a crucial role in traceability by offering a detailed audit trail of transactions on a network. Most of the existing works exploring the application of blockchain for traceability are in the context of supply chain management [113–116]. Supply chains generally have a complex network of manufacturers, suppliers, retailers, distributors, auditors, and customers. Smart contracts record the movement of assets across these entities in the blockchain in an immutable and tamper-proof manner. The shared ledger infrastructure enables anyone to retrieve the audit trail to find the different milestones an asset has crossed in the journey through the supply chain. Unlike physical commodities in the supply chain that are only owned by one specific entity, data ownership can include multiple parties as long as they can access the data and share it with others [117]. Several studies [118, 119] employed blockchain to enable provenance and lineage traceability. A data sharing scheme is introduced in [120] based on the blockchain double-chain structure to separate the original data storage and the transactions. One chain was used to store the encrypted data, and another chain was used to record the data-sharing transactions. These signed data-sharing transaction records enable data to be traced. In another approach [121], blockchain was applied to an m-health system for secure patient-user access and traceability of electronic health records. This implementation is based on Hyperledger Fabric which enables the creation of chronologically organized and immutable health data records traceable throughout the system, maintaining the necessary anonymity. The proposed system implements two separate database components (personal and health data) that assure data traceability through sets of IDs stored in the blockchain to create this anonymous storage system.

Recall from Chapter 1 data ownership traceability in the data marketplace becomes more challenging since a buyer can buy data in one marketplace and sell it to participants in another. The existing approaches discussed above enable data traceability within the confines of a single system and do not consider the dispersal of data ownership spanning across multiple systems. Moreover, a buyer can redistribute/resell data without the knowledge and consent of the provider. Such undisclosed reselling can lead to privacy violations or monetary loss for the providers. Managing digital data rights is fundamental to restricting buyers' ability to redistribute/resell the data they have purchased. A data owner should be fairly compensated if buyers resell their data to others in the marketplace. In the next section, we first discuss the existing interoperability mechanisms. Next, we provide an overview of the digital rights management (DRM) systems followed by the fair payment settlement in data trading. Lastly, we present a comparative summary of the most relevant state-of-the-art with our proposed approach.

Interoperability mechanism: Tracking data ownership spanning across multiple marketplace systems requires these disparate systems to interact and integrate seamlessly, allowing for data sharing between them. In this section, we will discuss several existing blockchain platforms that aim to achieve interoperability between different blockchain networks.

ChainBridge [122] is a protocol that enables the transfer of digital assets and tokens across different blockchain networks through relay chains and parachains, each with their own bridge contract. The communication between different chains is through a message-passing system. To transfer assets, the originating chain's bridge contract creates a message and sends it to the target chain's bridge contract. However, it requires a trusted set of validators to maintain the relays, making it not fully decentralized.

Polkadot [123] uses a multichain architecture with "parachains" that run in parallel to the Polkadot Relay Chain. Each parachain is a sovereign blockchain with its own governance structure and token economics. The Relay Chain acts as a bridge between these parachains, enabling them to communicate and share data and assets using a cross-chain message passing protocol. However, running a parachain on Polkadot requires significant resources, which may limit participation, and the Relay Chain may become a bottleneck if it cannot handle increased traffic from the parachains.

Cosmos [124] uses a hub-and-spoke architecture to create multiple independent blockchains or zones that communicate through the Cosmos Hub. Each zone can communicate with other zones using the Cosmos Inter-Blockchain Communication protocol. However, the mechanism requires high coordination between zones, limiting scalability. Also, the Cosmos Hub may become a central point of failure if compromised or offline.

Avalanche [125] uses subnets and a bridging mechanism to enable interoperability between blockchain networks. X-Chain is used for asset exchange between blockchain networks and C-Chain is used for smart contract execution and development. These subnets are bridged using the Avalanche-Ethereum Bridge. However, larger subnets may have more power and influence over the network, which could lead to centralization risks.

Cardano [126] uses a two-layer architecture: a settlement layer for accounting and a computation layer for smart contracts and dApps. Its cross-chain communication protocol enables the transfer of assets and data between different blockchains, even those not built on Cardano. However, these features are still in development and yet to be fully tested.

Algorand [127] utilizes Atomic Transfer Protocol for decentralized cross-chain asset transfers. This involves hashing transaction data from one blockchain and including it in another blockchain's transaction, ensuring atomic transfers. Additionally, Algorand offers layer-1 smart contracts using TEAL programming language. However, low adoption of Algorand may limit the availability of resources and tools for developers.

Polygon [128] uses Plasma Layer 2 scaling solution to address Ethereum's scalability and usability issues. Its architecture includes interconnected sidechains that are compatible with Ethereum Virtual Machine networks. Polygon enables interoperability with non-EVM compatible networks via its Polygon Bridge. However, currently, it only supports Ethereum-based networks.

IBM's blockchain interoperability mechanism [129] uses a trusted data transfer protocol based on relay services and attestation proofs, focusing on data integrity and security. Each blockchain network has a relay node that receives and verifies data from one network and transfers it to the other. Attestation proofs are cryptographic signatures that are created by trusted parties or validators on each network. The relay nodes collect these attestation proofs from both networks and use them to verify the authenticity of the data and prevent tampering by malicious actors during the transfer process.

In this thesis, we opted for IBM relay and attestation proofs based interoperability mechanism due to its ease of implementation, flexibility, and compatibility with various blockchain networks. Its focus on data integrity and security further supports our goal of facilitating trusted data transfer across multiple marketplace systems. However, the relay nodes may introduce centralization and trust issues that we address in our research.

Digital rights management (DRM) systems : To date, several works have been proposed based on digital watermarking and blockchain for managing digital data rights, which is essential for data trading. DRM controls the trading, protection, monitoring, modification, distribution, and tracking of digital data [130].

The authors in [131] and [132] propose a distributed media transaction framework that uses blockchain's append-only and time-ordered property to record the modification history and ownership of an image. A digital watermark comprising the corresponding transaction ID and other information such as copyright owner, location, and creation date is embedded in the image to link the transaction trail recorded in the blockchain with the image. In other similar approaches [133] and [134], the authors present a copyright management model for digital images using digital watermarking, blockchain and IPFS. Blockchain stores information about the digital image, such as the owner's digital signatures and cryptographic hashes. The watermarked image and corresponding block index are stored in IPFS. The above schemes provide a reliable, secure, efficient and tamper-resistant digital content service and DRM practice. However, the transaction trail can only be retrieved by replaying all transactions, which requires traversing every transaction in every block, which is time-consuming and slow. Moreover, this retrieval approach is applicable for offline analysis and unsuitable for on-chain transaction processing. In other words, smart contracts cannot access historical transaction trails, hence, restricting the expressiveness of the business logic that the contract can encode.

Another popular use case of ownership traceability is non-fungible tokens (NFTs) [135]. NFTs are digital assets that represent ownership of unique real-world items such as art, music, and videos. When creating a new NFT, a smart contract is deployed on the blockchain that contains the NFT's metadata, including its unique token identifier, ownership information, and transaction history. The transfer of an NFT is transparent, irreversible, and recorded on the blockchain, providing an immutable record of ownership transfers. However, NFT smart contracts only record the current owner and associated metadata, and do not provide the complete ownership trail, which is needed in data marketplaces to enable payment sharing among all owners. To obtain the entire ownership history of an NFT, one needs to examine the input and output addresses associated with each transaction, which can be a time-consuming process. Additionally, NFTs can be transferred between compatible blockchain networks, enabling ownership traceability across these networks. However, these networks should be compatible with similar token standards, or it may not be possible to transfer the NFT to those networks.

DRMChain [136] is a digital copyright management that ensures authorized access to digital content and provides external storage of decentralized digital content using IPFS. DRMchain employs two isolated blockchain application interfaces for storing the original content with its cipher summary and the DRM-protected content service, such as content watermark, encryption, license and violation tracing, among others. DRMchain enables violation tracking by tracing the identity of the content provider responsible for the illegal content. However, the system lacks traceability functions such as trail retrieval or validation.

In [137], the authors provide a data traceability model addressing the problem of limited storage resources of edge nodes that cannot store the blockchain. The approach provides a secure data traceability solution by dividing the edge network into multiple blockchain networks. The nodes in the edge network are divided into groups known as internal areas based on their physical distance from each other. For each internal area, a master node is elected that has the highest computing power and forms a peer-to-peer blockchain network, also known as the external area, with other master nodes. The internal area uses digital watermarking for data distribution and traceability. For data transmission outside the area, master nodes store the transmission record and the watermark information on the blockchain network. However, tracing the path along which data is exchanged requires querying the blockchain, which takes time and is thus not suitable for online settlement of resell payment sharing.

The authors in [138] propose a digital watermark management system based on smart contracts to prevent the illegal distribution of publications. Their solution uses a special deployment method for computationally intensive contracts (CICs) that enables smart contracts to perform computationally intensive calculations such as digital watermarks and complex encryption algorithms at low gas costs on Ethereum. Besides the low technology readiness of CICs, the approach could suffer from scalability issues as it stores the data in the ledger and implements computationally expensive watermark embed/detection algorithms using smart contracts.

In [139], the authors propose a watermarking technique for big data by leveraging the power of blockchain technology and smart contracts. An owner stores the watermarked data in IPFS and registers it to the system. Data is transferred to data collectors using the proxy re-encryption technique and tokenization. An access control mechanism is devised to ensure that only legitimate buyers can access the data after getting permission from sellers. The system provides an audit trail for data movement for legitimate data trading. However, the model does not detect the unauthorized spread of the divulged copies across marketplace systems. Moreover, the system lacks diversified trade trail management functions, such as trail validation or retrieval. In addition, it lacks an effective resell payment-sharing mechanism.

Payment Settlement in Marketplace : In our framework, we consider decentralized data trading and resell payment share settlement among data owners using smart contracts. This section presents the related work in this domain.

The authors in [140] address the issue of data ownership and identity verification during dynamic data streaming trading based on publish/subscribe model and IoT-brokered infrastructure. The data stream is stored on the Tangle and transmitted through Masked Authenticated messaging (MAM). Brokers manage the provider's product and corresponding subscribers using smart contracts. Smart contracts are also used to automate the subscription and payment process. Nevertheless, the above work does not consider ownership traceability and reselling of data. Since their model relies on brokers to certify transactions before uploading to the contracts, their model may suffer from collusion attacks.

The authors in [141] propose a data trading scheme using a special Bitcoin script for fair and instant payment. The scheme prevents double-spending attacks in fast transactions through ECDSA signature vulnerabilities. If the signature uses the same random number twice, the private key of the payer's signature is exposed, and miners set all of the Bitcoin of the payer's payment account to the transaction fee. The scheme allows sellers to upload their data securely on the cloud and stores integrity proof of data in the blockchain for validation purposes. Their approach focuses on instant payment using Bitcoin and prevents illegal data access through an access policy. However, there is no mechanism to detect reselling of data.

In [142], the authors propose three traceability solutions (Trace-MIN, Trace-MAX, and Trace-BF) to enable monetization for the IoT data marketplace. In Trace-MIN, only the broker has write access, and the information recorded on the blockchain is minimal. In Trace-MAX, maximum information is logged in the blockchain. In Trace-BF, three bloom filters are used for storing publication, tracking and confirmation information by the publisher, broker and subscriber. For each solution, authors present data sharing, verification and monetization process. Their approach employs a smart contract for keeping the count of data bytes exchanged between buyer and seller for payment settlement purposes. However, Trace-BF does not provide traceability of the data ownership.

In [143], a smart-contract-based protocol is proposed to ensure that the provider is paid when consumers resell their data. A tripartite contract involving the provider, the reseller and the buyer is formed and stored on the blockchain. When the buyer pays for the item, the revenue is shared between the reseller and provider using a smart contract.

Reselling can be detected by employing a verification process that the buyer performs. A buyer broadcasts its verification information to all the sellers in the market. A genuine provider matches the verification information with the existing data items and determines whether the reselling is authorized. The difference between this model and our work is that the former limits the resell payment share only between the reseller and the provider. However, our solution can share the payment among all the data owners in the trade lineage. In addition, their work has not addressed the scenario of sharing across multiple marketplaces.

The above-discussed existing solutions are summarized in Table 2.3. We compare the existing solutions against the following dimensions: application domain for the traceability mechanism, its specific use case, providing a mechanism to detect the undisclosed resharing/ redistribution, verifying the data origin and integrity, supporting traceability across multi-system networks, enabling distribution of resell payment. As is evident, most existing approaches detect undisclosed resharing or redistribution of assets and provide a mechanism to verify the data origin and integrity using watermarking techniques. However, they do not ensure traceability outside the confines of the system. In the data marketplace, participants will likely register with multiple marketplaces for monetary gain or alternative offerings for data purchases. Users could buy data from one marketplace and resell the original data or value-added services to users registered in other marketplaces. Furthermore, there is also a lack of consideration to protect the profit of data owners when their data is resold.

In Chapter 5, we propose TrailChain addressing the aforementioned challenges to devise an effective data ownership traceability in the multi-marketplace data trading scenario. In particular, TrailChain fulfils the following requirements: (a) provides a data ownership registration mechanism that establishes trust in the origin and creation process of data by guaranteeing that the registered data in the marketplace is indeed collected by the specific provider's IoT device, (b) empowers data-owners to track the data movement within and across marketplace systems, (c) provides an automated mechanism that can verify if the data was transferred without violating data owner's consent when the data is resold/redistributed, and (d) retrieves the trusted trade trail in a transparent, efficient and autonomous manner for enabling fair and protected distribution of the resell payment among the data-owners.

Table 2.3: Comparison of Traceability Approaches in Decentralized IoT Data Marketplace

Article	**Year**	**Application domain**	**Use-case**	**Undisclosed re-sharing**	**Verify authenticity**	**Multi-system**	**Resell payment**
[120]	2018	Data sharing	Modification history and ownership of data	✓	✓	✕	✕
[121]	2022	Health system	Health records	✓	✓	✕	✕
[131], [132]	2017, 2020	Distributed media transaction	Modification history and ownership of media	✓	✓	✕	✕
[133], [134]	2018, 2020	Copyright management model	Ownership of digital images	✓	✕	✕	✕
[136]	2018	Digital copyright management	Identity of the content provider	✓	✕	✕	✕
[137]	2020	Data traceability model	Modification history and ownership of data	✕	✕	✓	✕
[138]	2019	Digital watermark system	Illegal distribution of publications	✓	✕	✕	✕
[139]	2020	Data marketplace	Ownership traceability	✓	✓	✕	✕
[140]	2020	Data marketplace	None	✕	✓	✕	✕
[141]	2020	Data marketplace	None	✕	✓	✕	✕
[142]	2020	Data marketplace	Trade traceability	✕	✕	✕	✕
[143]	2020	Data marketplace	Data originator	✓	✓	✕	✓

2.4 Chapter Summary

In this chapter, we first presented a brief overview of blockchain technology and discussed the structure of transactions and blocks, and key components of blockchain infrastructure. We outlined some popular blockchain platforms and examined their features. We presented an overview of the key components of the IoT data marketplace. Then we discussed the literature in the context of blockchain solutions addressing autonomy, efficiency, privacy and traceability challenges in the IoT data marketplace, covering industry-led and academic solutions. We grouped the existing works based on their approach to architecting blockchain solutions for achieving autonomy and efficiency in the decentralized IoT data marketplace. In particular, these approaches include trade transaction management, data catalogue, hybrid centralized-decentralized model, data storage and access control and agreement instantiation. For the privacy challenge, we discussed the existing work of privacy-enhancing technologies, privacy-aware frameworks and label/rating and scoring systems. We looked into the literature on digital rights management systems and payment settlement for traceability challenges. Finally, we compared the limitations in the discussed approaches to our proposed solutions which are explained in detail from Chapter 3 to Chapter 5. At the end of each section, we provided a comparative analysis of the discussed articles using tables.

Chapter 3

MartChain: Towards an Autonomous and Efficient Decentralized IoT Data Marketplace

In this chapter, we answer the research questions **RQ1**, **RQ2**, **RQ3**, and **IQ**. The unprecedented rate of IoT adoption presents an opportunity for device owners to trade their IoT data with interested buyers. Traditional solutions rely on a trusted third party for conducting all management operations, thus raising issues such as centralized governance, single point of failure, limited user control and lack of transparency. Blockchain is a promising technology that can make data trading decentralized, secure, democratic and transparent. However, adopting blockchain in a marketplace ecosystem is not straightforward and requires consideration of the following aspects. First, given the large number of IoT devices owned by a myriad of providers, trading data generated by them may lead to a large volume of operational transactions. Thus, using a monolithic blockchain to record all such transactions will be inefficient, expensive and unscalable. Second, it is challenging to enable self-governance and non-repudiation without a trusted third party. Last, the restricted energy budget of IoT devices limits the volume of data they can exchange in real time. An inefficient allocation of buyers' demands on such devices poses a challenge for providers to simultaneously serve multiple buyers' demands without disrupting their service level agreements. In this chapter, we propose MartChain for trading IoT data in real time generated by resource-constrained IoT devices. MartChain employs facilitators to manage data offerings/demands listings and perform search and discovery. We leverage smart contracts to implement other key operational components such as contract management and execution, data pricing, payment and settlement, and reputation mechanisms. MartChain is underpinned by an Energy-aware Demand Selection and Allocation (EDSA) mechanism for optimally selecting and allocating buyers' demands on providers' IoT devices. We present a proof-of-concept implementation in Ethereum to demonstrate the feasibility of the framework. We investigate the impact of buyers' demands on the battery drainage of IoT devices under different scenarios through extensive simulations. Our results show that this approach is viable and benefits the provider and buyer by creating an autonomous and efficient marketplace.

3.1 Introduction

As discussed in Chapter 1, the Internet of Things (IoT) presents an enormous opportunity to transform society by unlocking and unleashing a world of data. As the number of data sources expands exponentially, businesses are investigating ways to harness the data for insights they can use to drive operational efficiencies, improve their customers' experience, or both. For example, a fitness tracking app provider may wish to procure air quality data from weather stations deployed all over the city to suggest pollution-free running tracks to its users. Grocery chains may be interested in obtaining aggregated information about food items stored in smart fridges of customers in a local neighbourhood to manage their inventory better. In traditional IoT systems (see Chapter 2), the data is stored by the service provider of a device leading to the creation of data silos where data access is only restricted to the service provider and the inability to share it with other parties. Given the significant value of the data produced by IoT devices, the notion of an IoT data marketplace has attracted tremendous attention. A data marketplace would enable IoT device owners to monetize, trade, and share the historical or real time data generated by their IoT devices with other participants in the IoT ecosystem. The data marketplace addresses the issue of data silos and incentivises users to share their data with service providers or other parties that require access to the data.

To date, data marketplaces for trading IoT data have typically been developed using a centralized brokered approach [21,24]. In such approaches, a TTP oversees the trade to ensure that both parties commit to their obligations, which results in several limitations. The first limitation was centralized governance, as the TTP conducts all management operations and controls every aspect of a trade, from offer listings to price determination, product search, data storage, transacting the trade, data dissemination, and buyer/provider feedback. With billions of connected IoT devices, it can be a bottleneck. Other limitations were limited privacy as the TTP has full visibility on the trade history of providers and buyers and the associated data, central point of failure, and low security as the TTP may be compromised by hackers, which leads to data leakage. To address the outlined challenges, blockchain [144] is a promising technology that has the ability to disrupt the data marketplace by making trading decentralized, democratic, secure and transparent. It provides multidimensional benefits in the IoT data marketplace as discussed in Chapter 1. Nevertheless, the adoption of blockchain to create an open IoT data marketplace is not straightforward and requires considerations from different perspectives, as listed below:

An open data marketplace refers to a market that is accessible to anyone to join and trade IoT data generated from IoT devices. Such a marketplace often spans the globe, where the number of participants may increase drastically. Therefore, before realizing the promise of blockchain for the IoT data marketplace, a fundamental challenge, i.e., scalability, must be addressed. With the ever-increasing number of IoT devices owned by a myriad of providers, generating a massive amount of IoT data leads to overwhelming transactions of read/write operations. Using one monolithic blockchain for storing all the marketplace-related transactions, such as all trade interactions, data offerings, and access policies on-chain, is expensive, inefficient and unscalable. An efficient IoT data marketplace must balance the degree of decentralization and scalability that a blockchain possesses without impacting the overall performance.

Moreover, in a decentralized data marketplace, ensuring autonomy and non-repudiation become challenging. An autonomous marketplace enables self-governance where participants define their own terms and conditions under which they will trade the data. Non-repudiation ensures that they adhere to those obligations. Consider an example where a buyer and provider agree to trade GPS data with specific sampling intervals, duration and quality at a certain price. However, ensuring that both parties adhere to these obligations is difficult. The provider may renounce the contract by not providing the GPS data for the specified requirement. Similarly, the buyer may not pay the agreed price for the purchased data. Such actions lead to agreement violations and impact the usability of the data marketplace. An autonomous decentralized data marketplace must ensure non-repudiation between parties without the need for any trusted intermediaries.

Another challenge in trading data streams from IoT devices is the associated energy budget, perhaps the most substantial factor limiting the volume of real time data. A data stream is generally quantified in terms of sampling rate, duration and quality. These parameters directly impact the energy budget of the IoT device. An IoT device's energy budget is subject to its battery capacity and, where applicable, to the energy it can harvest from its environment. Without energy harvesting, IoT devices must optimize their operation and adapt to their environments to deliver maximum utility. When energy harvesting is available, IoT devices have a dynamic energy budget subject to their physical context, current draw and prospective energy [145]. In order to be viable in the data marketplace, an IoT device must serve its primary purpose without impacting the user's experience. Besides, it should also serve the secondary purpose of fulfilling buyers' demands based on the devices' current residual energy and capabilities. This requires the provider to efficiently use the energy budget of their IoT devices so that buyers' demands do not deplete their energy and disrupt the data service. Disrupted data service will affect the provider's motive to maximize revenue and the buyer's goal to get quality data for the desired rate and duration. Therefore, it is essential that the selection and allocation of buyers' demands on the provider's devices be energy efficient.

Existing works on blockchain applications offer specific functionalities in the broader context of data marketplaces. For instance, in [53], blockchain is used as a trade transaction management system. In [11], blockchain is used by providers as a product catalogue that buyers can browse to find the proper data type based on their requirements. In another approach [54], blockchain smart contract is used to implement an access control mechanism enabling the providers to manage who can access their data and for how long. Although these approaches allow the sharing of IoT data, they do not have all the essential functions of a marketplace. A fully-functional and effective marketplace must support all the fundamental components for buying/selling activities [51], including user registration and authorization, data discovery and selection, price model, contract management, reputation management, data metering, billing and payments, real time middleware (see Chapter 2).

Table 3.1 compares existing blockchain-based marketplace frameworks with our proposed work. The main criteria for this comparison are the supported functionalities and the application of smart contracts in realizing those functionalities. This comparison demonstrates the need for a holistic and effective marketplace design with key functionalities that satisfy all the aforementioned complex requirements. Our proposed framework, MartChain, employs smart contracts to manage users, contracts, data price and reputation. It supports the following marketplace functionalities: (1) user registration to participate in data trade, (2) 2-step selection and matching of buyers' demands with provider's offerings based on available battery in their devices, (3) dynamic data pricing to encourage providers to provide high-quality data (4) formation of customizable and flexible agreement based on users' specifications (5) tuning of reputation scores based on the malicious behaviour of dishonest actors (6) data metering to facilitate settlement and dispute resolution (7) automatic payment and settlement at predetermined intervals (8) transfer data over a secured TCP connection.

In this chapter, we present MartChain, a hybrid data marketplace framework that uses facilitators to address the aforementioned scalability issue. Facilitators coordinate trading between providers and buyers by managing data offering/query listing and identifying matches from billions of devices based on buyers' requirements. Smart contracts are self-verifying, self-enforcing and tamper-proof and can thus automate the execution flow of the

Table 3.1: Comparison of blockchain-based data marketplace solutions with MartChain

Articles	**Smart contract application**	**User registration**	**Discovery and selection**	**Price Model**	**Contract management**	**Reputation management**	**Data metering**	**Billing and payments**	**Real time middleware**
[35]	Access control	×	×	Fixed	×	×	×	✓	×
[65]	Access control	×	DHT	×	×	×	×	×	×
[53]	Data offering, Trade agreement, Data receipt	×	Browse blockchain	Fixed	✓	×	✓	✓	×
[37]	Registration, Data offering	✓	Browse blockchain	Fixed	×	×	×	×	×
[11]	Data offering	×	Browse blockchain	Fixed	×	Rate data quality	×	SDPP [70]	✓
[54]	Access control	×	×	Fixed	✓	×	×	✓	✓
[146]	Access control, Data integrity	✓	×	×	×	Rate data quality	×	✓	✓
MartChain	User, Contract, Price and Reputation management	✓	2-step	Dynamic	✓	Rate experience	✓	✓	✓

trading whenever the predefined conditions are met. MartChain leverages smart contracts to develop operational functionalities. A new smart contract, *SubscriptionSc*, is deployed for each agreement between buyer-provider pairs. These contracts are customizable based on user specifications, maintain agreement integrity, enable autonomous data trading, and ensure non-repudiation. MartChain also includes a price model and reputation mechanism to enable other fundamental functionalities of the marketplace. Several price strategies were proposed in literature based on optimization problem formulation to achieve higher revenue [147–149] or game theoretic model based on buyer's demand [150]. However, these price schemes are complex, time-consuming and difficult to integrate with blockchain because of high resource consumption. As highlighted in Table 3.1, most of the existing frameworks utilized a fixed model based on the price specified by the provider. On the contrary, we employ smart contract, *PriceSc*, to implement a simplistic and convenient estimation of data price based on the market dynamics that motivate providers to supply high data quality in return for higher incentives. Furthermore, trust and reputation management is pervasive in the data marketplace, enabling trustworthiness among users who do not trust each other. As listed in Table 3.1, some existing marketplaces employ a reputation system to rate providers based on the quality of their offered data items. In [29], reputation scores are calculated based on the positive and negative activity of the user. However, this model suffers from manipulation of reputation score by various malicious activities like collusion, false negative feedback etc. Another existing trust and reputation model in literature explores the notion of trust in supply chain [151] or access control [92].

However, these works mostly overlook the trust issues amongst the users in the data marketplace. MartChain leverages smart contract, *ReputationSc* to assess the reputation of users based on their trading interactions and experience with each other. The flow control of all these interdependent smart contracts needs to be designed carefully to provide the correct workflow for each trading stage. Moreover, creating a new contract on demand can result in an increased number of interconnected contracts over time. Therefore, managing the interactions of these contracts becomes challenging. An optimization mechanism also underpins MartChain for selecting and allocating buyers' demands on providers' devices to support real time data streaming from energy-constrained IoT devices.

This chapter makes the following contributions:

- We present details of the blockchain-enabled marketplace framework, including the architectural design, the participants, and their interactions.
- We formulate the demand selection and allocation optimization problem to maximize the provider's revenue by considering the resource-constrained nature of IoT devices and solving it using our novel greedy heuristic algorithm. Simulation results confirm that providers and buyers will benefit from the proposed algorithm.
- We implement critical components of the marketplace, such as the agreement framework, pricing model and reputation model using smart contracts and present a proof-of-concept implementation of our framework in the Ethereum testnet to demonstrate its feasibility.

The rest of this chapter is organized as follows. The proposed framework and network architecture are discussed in section 3.2. In section 3.3, we present an integer linear formulation of the optimization problem and a heuristic algorithm to solve the problem, followed by the details of our smart contracts in section 3.4. We present proof-of-concept implementation and evaluations in section 3.5, and section 3.6 summarizes the chapter.

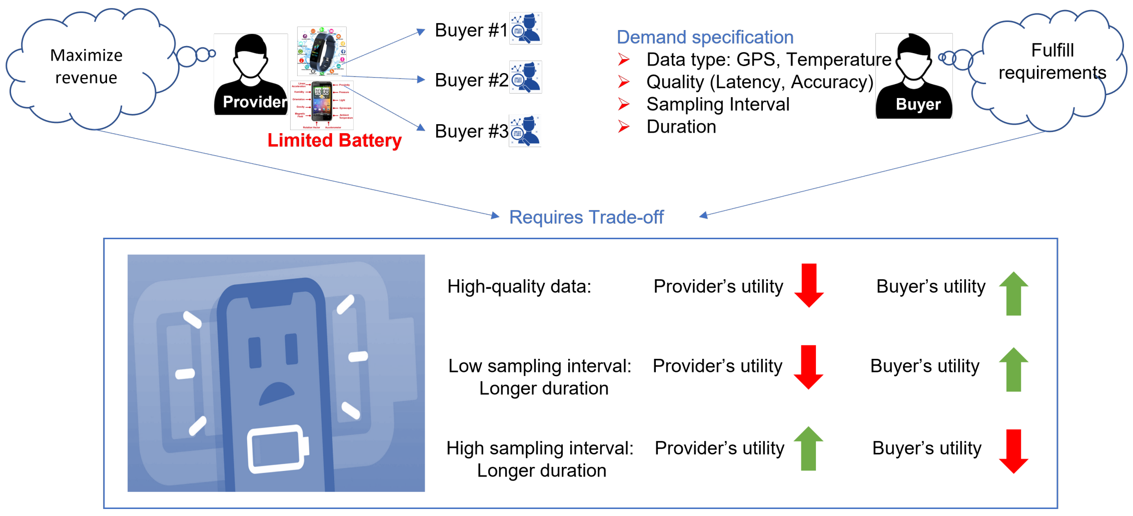


Figure 3.1: Motivating use-case

3.2 Proposed MartChain Framework

In this section, we first provide a motivating use case followed by a high-level overview of MartChain and then discuss the major components of the framework.

3.2.1 Motivating Use case

A motivating use case is depicted in Fig. 3.1. Consider an application developer providing location-based services by purchasing real time location data from multiple providers. The developer specifies their desired data quality in terms of accuracy and latency, sampling interval, and duration. These parameters directly impact the battery consumption of the provider's device. Existing techniques for improving the power consumption of IoT devices usually compromise data quality for energy efficiency. High-quality data (high accuracy and frequency, low latency) usually consumes more energy. For instance, the positioning modality that provides the most accurate location information and power consumption is listed in Table 3.2.

Similarly, when an IoT sensor works at a low sampling interval and for a longer duration, it imposes a heavy workload on different components (processor, network, storage, and

Table 3.2: Power consumption and location accuracy of different positioning modality

Position modality	**Location accuracy**	**Power consumption**
GPS	6m	High
Assisted GPS	60m	Medium
Cell-ID/WiFi Positioning System	1600m	Low

sensors) of the device, draining the battery rapidly. For example, if a smartphone owner sells location data by sending frequent (low sampling interval) data, the phone battery depletes more quickly. This degrades the smartphone user experience and impacts the utility of other buyers with whom the provider has already made an agreement. However, if a low data rate setting is applied, then the data quality is reduced, and the buyer's utility is degraded. Due to the costs associated with continuously sensing and transmitting data, a provider has to make an optimal decision to serve buyers' demands in real time based on the devices' current residual energy and capabilities. Thus, finding a solution that maximizes the participants' satisfaction while considering IoT devices' limited capabilities and constraints for creating a sustainable marketplace is vital. With this use case in mind, we present the MartChain framework next.

3.2.2 Overview

The architectural design for MartChain is shown in Fig. 3.2. MarChain is a 2-tiered marketplace framework comprising facilitators in tier 1 and participants, including buyers, providers and their devices in tier 2. Instead of a centralized trusted facilitator, we leverage the concept of geographically distributed facilitators in tier 1 that are interconnected in a P2P manner, thus forming an overlay network spanning the entire globe. Tier 2 is partitioned into multiple service zones that the facilitators supervise. We assume facilitators to be trustless and highly resource entities. They can be realized using fog nodes to ease the burden on resource-constrained IoT devices. MartChain incorporates a fee-based mechanism that we assume will motivate independent, self-interested facilitators to participate in the marketplace. To balance the degree of decentralization and scalability a blockchain possesses, we carefully split the marketplace functionalities be-

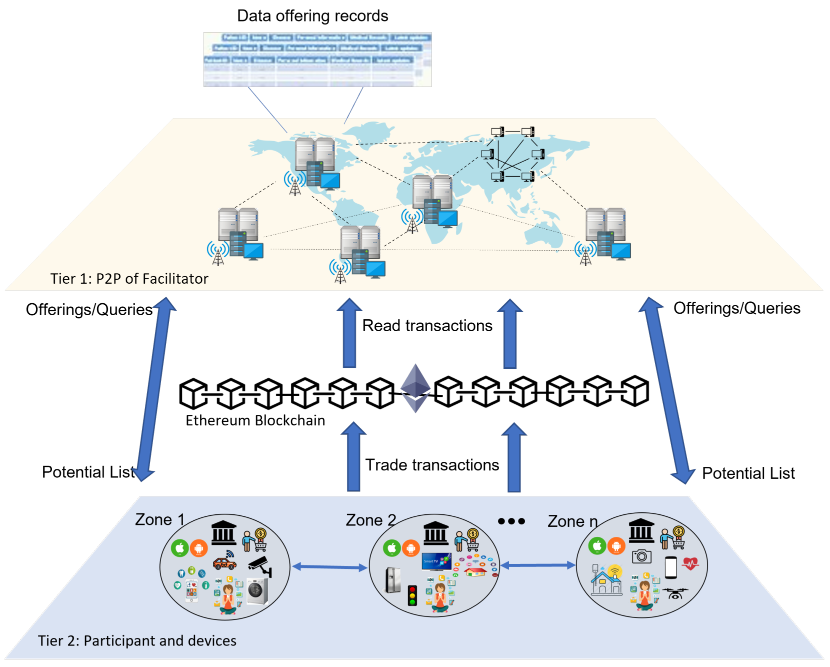


Figure 3.2: 2-tier framework of MartChain

tween the participants in tier 1 and tier 2. The participants in tier 2 must register with a facilitator nearest their location and create a profile. The facilitator gathers all the data offerings and query lists from all the registered providers and buyers in their service area respectively, executes search and match algorithms and provides them with the potential buyer and provider lists. The resource-rich facilitators can easily handle the processing overhead of searching and matching. Since facilitators are trustless, they can collude with any participant and provide a biased potential list. Such malicious behaviour can be prevented by using decentralized database systems such as BigChainDB or blockchain-based data catalogue approach (see Chapter 2) to maintain the device and data information transparently and efficiently. This enables a decentralized scheme for handling product information and further optimizes transaction requests to deal solely with product data references rather than trading transactions. Since the facilitators are synchronized, if a facilitator fails, the associated participants connect to another nearby facilitator. Furthermore, to eliminate the facilitator's monopoly market power over how providers are selected and advertised to buyers, the generated lists can be validated and attested by other facilitators in the network using attestation-based proof [129]. The second tier-segregates transaction requests dealing with references to the product/query data from the trading transaction. The trading-related operational transactions are handled by a decentralized marketplace application (Dapp) built over a public Ethereum network that employs all the other key marketplace components, such as managing contracts, pricing, payment and settlement, reputation mechanisms, etc. These are implemented using smart contracts to enable autonomous IoT data trading as discussed in section 3.2.3.

The main roles of different participants in MartChain are explained below.

Provider - Providers mainly include IoT device owners interested in selling their data. A provider can have multiple heterogeneous devices with varying resources and data characteristics. To connect these devices to the network, a conceptual data repository such as Databox [152], Personal Data Stores (PDS) [153] or Solid [154] can be used. IPFS [155], Swarm [156] or Web3.0 [157] are prominent examples of personal data management via such data stores that enable user-controlled data sharing. These technologies are peer-to-peer (P2P) with decentralized file transfer systems in which files are addressed by the hash of their content. Moreover, they are compatible with edge computing. IPFS or Swarm nodes can be executed on IoT gateways connecting local IoT devices to the P2P network. Even for IoT devices that are not co-located, these nodes can aggregate and store the data of all the connected devices via the Internet. The provider posts the offerings to the facilitator as metadata. The metadata consists of information such as a public identifier (i.e., changeable public key) to anonymize the identity of the provider; a list of IoT devices owned; geographical co-ordinates associated with the sensing data; spatial and temporal context of the data which can be static or dynamic; and a data usage license which defines the appropriate use of the data and also includes terms for restrictions of reselling of the data. We address the problem associated with data reselling by unauthorized agents in Chapter 5. We assume that a provider uses two strategies to generate a sustainable income in the marketplace: (1) maximizing revenue by using the available resources on IoT devices as efficiently as possible and (2) attracting buyers by offering add-on features such as high-quality IoT data or sensitive data which are distorting, revealing or intruding their privacy.

Buyer - Buyers are the end users interested in buying the raw data or value-added service. They can be private or government organizations or individual users. We assume that a buyer can also act as a data reseller (see Chapter 5) depending on its activities in the marketplace. A buyer queries the system defining the required data and its specifications. A query list generally includes buyer's requirements, which are classified into context-based and quantitative-based. In a query, context-based requirements comprise data type, location, temporal context (i.e. archived or real time data), and provider's reputation score, and quantitative-based requirements consist of data quality, sampling interval and duration. The difference between the two is that the latter depends on the battery availability of the provider's devices, while the former is independent of the available battery. Queries that include location information can support location-based value-added services or limit the search for matching potential providers. For instance, a smart refrigerator manufacturer may need data from refrigerators of their customers in different geographical regions to gain insights into how their devices are being used. The manufacturer can query the marketplace to collect extensive usage data about their refrigerators, including temperature settings, number and frequency of door openings, the average number of items in the refrigerator, energy usage, location, etc.

Facilitator - A facilitator is a trustless and resource-available entity whose motivation is to receive incentives in return for its service. The facilitators serve the following purpose and benefits in our system:

- Catalog: A facilitator receives data queries from buyers and metadata for data offerings from providers. The facilitators maintain the list of the provider's data offerings and the buyer's demands that belong to their service zones. On receiving the queries from the buyers in their service zone, a facilitator multicasts their query list to all the other facilitators in the network so that the request can be routed to the appropriate facilitator based on the specified parameters.
- Search and match: The facilitator uses a discovery and selection algorithm; an example is illustrated in [158] to match the data offerings and demands based on the

criteria specified by both parties, such as location, data types, and budget pricing. Data streaming requires buyers to define quantitative parameters such as sampling interval, streaming duration, and data quality, along with contextual parameters such as data type, location, and provider's reputation score. The facilitator matches the metadata of data offerings to the contextual parameters of the buyer's demand. The selection based on quantitative requirements needs to consider the battery available on the provider's devices, which is a dynamic parameter. To avoid the network overhead related to periodically notifying facilitators of the providers' device battery status, selection based on quantitative requirements is made locally at the provider's end using the optimization module as explained in section 3.3. Next, the facilitator creates a list of potentially matching buyers and sends it to all the identified providers. It may be possible that multiple providers satisfy the query of a single buyer. A facilitator sends this one-to-many selected list to the corresponding buyer. The match may involve participants associated with other facilitators. However, it could also be possible that no match happens. In this case, the facilitator retains the query and data offerings and checks if a match is possible in the future.

- Dispute resolution: At the end of data trading, both the buyer and the provider submit the cumulative number of data samples transferred for settlement purposes. The provider and buyer are untrusted entities that may behave maliciously. Such malicious behaviour can be detected by the other party involved in the same trade and then is reported to the facilitator, who functions as a judge to resolve the dispute. The facilitator relies on the trade's claims recorded in the blockchain in tier 1 and evidence provided by either side to solve the dispute. MartChain can employ a proof of delivery solution as discussed in [159] to prove the trading and delivery of digital assets. However, a facilitator can collude with the participant to give a dishonest verdict. An affected participant can request the other facilitator, who can easily detect such collusion activity by referring to the trade transactions recorded in the blockchain. The participation of such dishonest facilitators is revoked from the network. Based on the trading history of a user, a reputation score is recorded in the blockchain, which signifies the probability that the user will behave honestly in the

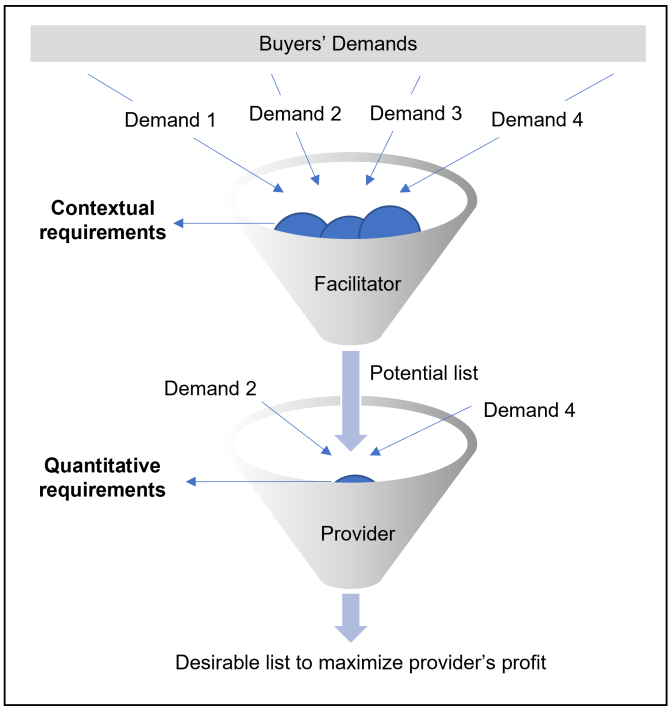


Figure 3.3: 2-steps demand selection and match

next transaction. In the case of detecting misbehaviour using proofs, the facilitator penalizes the responsible party, reduces its reputation score, and restrains the party from participating in the marketplace for a specific time period. If none of the parties is at fault and the dispute happens due to other reasons, such as channel congestion, no actions are being taken by the facilitator. However, the parties still need to pay a nominal fee to the facilitator for dispute resolution to prevent participants from abusing or misusing its service.

3.2.3 Main Components

Our marketplace framework supports the following main components:

Optimization-based selection and allocation: MartChain employs 2-step demand selection and match as depicted in Fig. 3.3. The first step of discovering and matching data offerings and queries is based on the contextual requirement of the demands (data type, location, and provider's reputation). It is actioned by the facilitator, as explained earlier. The second step is actioned at the provider, which involves optimization-based selection using quantitative requirements (sampling interval, quality, duration) of buyers' demands.

Agreement framework: An automated creation and enforcement of legally binding trade agreements is required throughout data transmission. Providers can specify their terms and conditions using smart contracts as an agreement instantiation. This allows them to define their product details (e.g., price) and all other content, such as subscription details, data quality, and data usage, in the way that best suits their needs. It also ensures that the participants' behaviour automatically conforms to the terms of the agreements. Smart contracts enable fine-grained per-user and per-data stream conditions to be formalized. The agreement framework consists of two smart contracts known as data subscription contract (*SubscriptionSc*) to record subscription details and register contract (*RegisterSc*) to record contract details as explained in sections 3.4.1 and 3.4.2, respectively.

Price model: We consider dynamic pricing, which depends on market conditions. The MartChain thus supports a competition-based price model that enables a simple and dynamic evaluation of IoT data price using the *PriceSc* (discussed in section 3.4.3).

Reputation Model: All the parties need to provide feedback to each other based on their trading experience to establish a trustworthy and reliable framework. Our reputation mechanism provides resilience to manipulation of reputation scores from various attacks such as ballot stuffing: the user gives false-negative feedback to one entity to make him quickly lose their reputation; strike and rechange: the user builds up a good reputation by performing many low-value transactions well and then misbehaves in a very high-value one; and collusion: two users collude to influence each other's reputation. The reputation model utilizes the *ReputationSc* (see section 3.4.4) to record their trading interactions in blockchain and evaluate the actor's reputation score. This ensures reliability among actors who do not trust each other.

The interactions of the above-stated components to achieve data trading are illustrated in

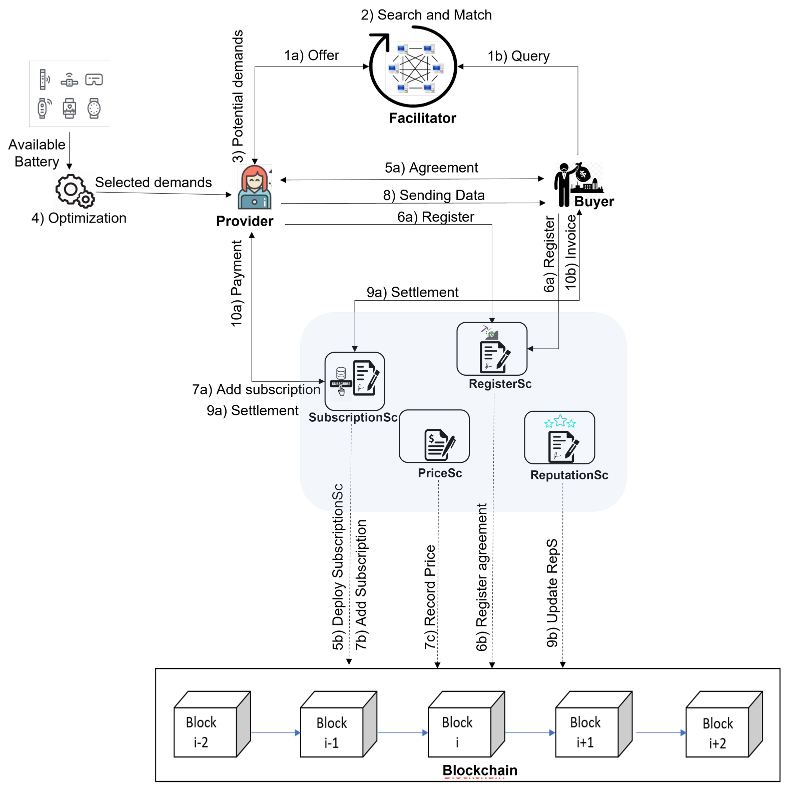


Figure 3.4: Interactions of different components in the MartChain

Fig. 3.4. The provider and buyer advertise the offer and query the facilitator, respectively. The facilitator matches the offer and query and sends the potential list of demands to the provider. The provider then uses the optimization module to select demands based on the available battery of their IoT devices. Subsequently, the provider and buyer come to an agreement. A *SubscriptionSc* is deployed in the blockchain and registered in the marketplace using the *RegisterSc*. The provider adds the subscription details in the blockchain using the *SubscriptionSc*. This updates the price of the requested data type using the *PriceSc*. The data is transferred off-chain from the provider to the buyer over a secured TCP connection. On completion of the data transfer, involved actors submit feedback that triggers a payment settlement. During settlement, payment is released to the provider, and an invoice is sent to the buyer. Lastly, the reputation scores of both actors are updated via a *ReputationSc*.

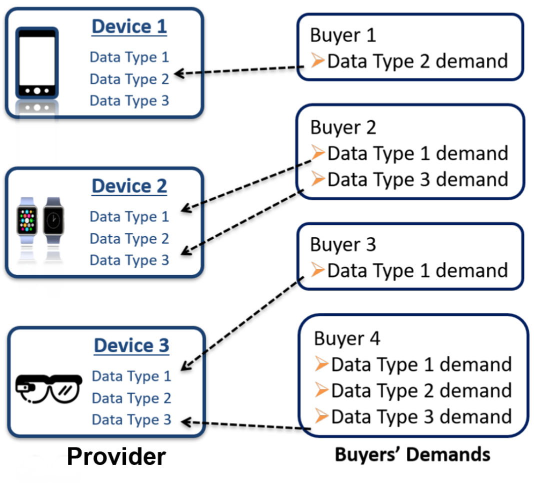


Figure 3.5: EDSA sample scenario

3.3 Optimization-Based Selection and Allocation

To fulfil buyers' demand for data-stream from resource-constrained IoT devices, MartChain proposes an optimization module. It allocates a subset of the demands on the provider's devices so that the provider's revenue is maximized. The selection and allocation of demands should meet the quantitative requirements of the selected buyers' demands and ensure that the battery capacity of the devices is sufficient to fulfil the assigned demands completely without any interruptions. This optimization problem is defined as the Energy-aware Demand Selection and Allocation (EDSA).

In this section, we formally state the EDSA problem. Given a set of demands where each demand is expressed in data type, sampling interval, duration, and quality, the aim is to find a subset of demands that maximize the provider's revenue while satisfying the battery, quality and allocation constraints. Fig. 3.5 describes the EDSA sample scenario for a provider with three devices offering three different data types and four buyers with different data demands. All the demands of $buyer_1$, $buyer_2$, and $buyer_3$ are selected and assigned to $device_1$, $device_2$, and $device_3$, respectively, while only one demand of $buyer_4$ is selected and assigned to $device_3$.

The EDSA can be formulated as an Integer Linear Program (ILP) to find the optimal selection of demands of the buyers to maximize the revenue as given in (3.1) through (3.4). The formulation uses the following variables that are categorized into two types:

1. *Provider's parameters:* A provider owns D resource-constrained devices with battery levels B_i, for $i = 1, 2, \ldots, D$. Each device provides a set of data types S. Without loss of generality, let Q_{ij} be the highest quality threshold that a device i can meet for a data type j. The unit data price $P_{ij}(q^k)$ is defined based on the device i, data type j and the quality demand q^k of the buyer k. The energy consumption E_{ij}^k of device i is the energy required for sensing, processing, and transmitting data type j to the buyer k.

2. *Buyer's parameters:* There are a total of C potential buyers, where each buyer may have multiple demands with different requirements of data types. A buyer k's demands consist of data type j, and the subscription duration is d_j^k with a sampling interval of s_j^k. Hence, $N_j^k = (d_j^k / s_j^k)$ is the total number of samples of data type j demanded by buyer k.

$$\underset{x_{ij}^k \in \{0,1\}}{\text{maximize}} \sum_i^D \sum_j^S \sum_k^C x_{ij}^k N_j^k P_{ij}(q^k) \tag{3.1}$$

subject to

$$\sum_k^C \sum_j^S x_{ij}^k E_{ij}^k \leq B_i, \quad \forall i, \tag{3.2}$$

$$\sum_i^D x_{ij}^k \leq 1, \quad \forall k, \forall j, \tag{3.3}$$

$$x_{ij}^k q^k \leq Q_{ij}, \quad \forall i, \forall j, \forall k \tag{3.4}$$

where the decision variables $x_{ij}^k = 1$ if data type j demand of buyer k is assigned to device i, and $x_{ij}^k = 0$ otherwise. Equation 3.2 captures the battery constraint of the devices. The allocation constraint in Equation 3.3 ensures that each buyer demand is served by at most one device of the provider. Equation 3.4 represents the devices' quality constraint, which defines the assignment restrictions. The integer linear programming formulation outputs a selection of demands that optimally maximizes the total revenue while meeting the battery, quality, and allocation constraints. The objective function is the total revenue generated by the demands that are selected and allocated to an IoT device of the provider.

The above problem can be mapped to a Multiple Knapsack Problem with Assignment Restriction (MKP-AR) [160]. The MKP-AR is defined as follows. Its input is a bipartite graph $G = (X, Y, E)$ with a set of edges E between X and Y. The vertices of $X = x_1, x_2, \ldots, x_m$ correspond to knapsacks (providers' devices). The vertices of $Y = y_1, y_2, \ldots, y_n$ correspond to items (demands) where m is the total number of providers' devices and n is the total number of demands. E is defined between (x, y) if quality Q_i offered by device x_i is greater than the quality q_j demanded in demand y_j. Item $y \in Y$ is assignable to knapsack $x \in X$ only if $(x, y) \in E$ (assignment restriction). For each knapsack $x_i \in X$, its capacity c_i (battery capacity) is associated. For each item, $y_j \in Y$, the profit (price) and the weight (power consumption) of y_j, denoted by p_j and w_j, respectively, are associated. A feasible solution of MKP-AR is assigning items to knapsacks such that, for each x_i, the total size of assigned items to knapsack x_i is at most c_i. The goal of the MKP-AR is to maximize the total profit of assigned items. MKP-AR is considered in [160] in which the profit and weight of each item are the same. A more generic problem, in which profits and weights are different, was studied by [161]. It uses linear relaxation to find optimal vertex solutions followed by solving an instance of reduced MKP-AR problem iteratively. MKP-AR is NP-hard in the strong sense, and no simple method to run on resource-constrained devices, such as dynamic programming, is known to apply to the MKP-AR. Also, given the low latency demand of a marketplace, computing an optimal solution may not be ideal. Therefore, we use a modified greedy heuristic approach [162] to include quality constraints to solve the EDSA. The intuition behind our algorithm is that selecting demands with high prices and low energy consumption may result in high total revenue for the provider under energy constraints. Based on this intuition, we define the *Normalized Revenue* (NR) of demand as the ratio of the generated revenue to the energy consumption of the demand. If the prices are chosen according to the costs, this scheme also maximizes the provider's total profit. Our NR-based Algorithm 1 starts by initializing the list of demands (Line 1) and calculating the NR values for each demand in the list (Lines 2-4). Then, the list of demands is sorted in descending order of NR values (Lines 5-9). Note that demands with equal NR values are further sorted based on their prices. The rest of the algorithm (Lines 11-19) assigns demands to devices using the

Algorithm 1 NR-based algorithm executed by the provider

```
1: Initialization of the Demand list.
2: for each dem in Demand
3:    Calculate NR
4: end for
5: Sort demands in descending order of NR
6:    if elements in NR are equal:
7:       Sort demands in descending order of prices
8:    end if
9: end sort
10: Sort devices in ascending order of battery availability
11: for each dem in Demand
12:    for each dv in device
13:       Compare Device battery $\geq$ demanded energy
14:          Compare Device quality $\geq$ demand quality
15:             Assign dem to dv
16:             Update the available battery of dv
17:             break
18:    end for
19: end for
```

sorted demands list (starting from the demand with the highest NR value), checking the battery availability and device quality constraints and updating the available battery of the devices.

3.4 Marketplace Components Using Smart Contracts

MartChain realizes critical components of the marketplace by leveraging smart contracts. There are two types of blockchain accounts: external and smart contract accounts. Once the actors register with the facilitator in their service zone, they receive a changeable key pair to anonymize their identity as suggested in [27] and become the external account holder. The private key is used to sign a transaction, while the public key is used to validate the transaction's signature. The smart contract is compiled into bytecode and deployed in the blockchain with unique addresses that can be called by any external account holder using a transaction or another contract using a message. A smart contract can consist

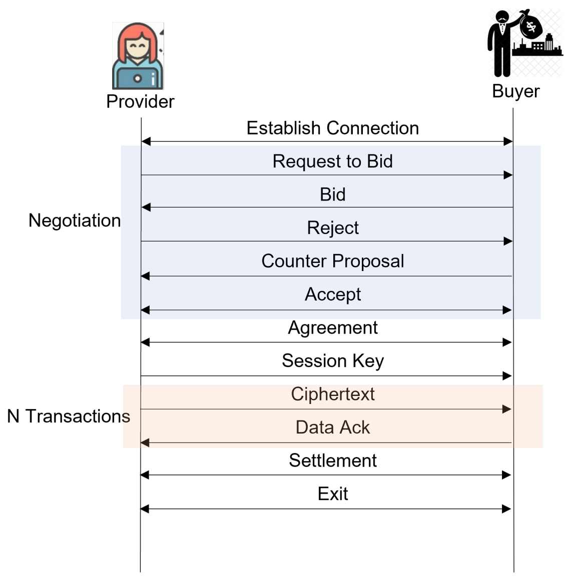


Figure 3.6: Data trading protocol

of several functions. So, an ABI is required to specify which function in the contract to invoke. A transaction specifies the contract address, and ABI triggers the function in the contract, which executes itself according to the coded terms.

After receiving the probable buyers' demands list from facilitators, the provider identifies the desired list of demands using the NR-based algorithm and runs the data trading protocol as shown in Fig. 3.6. Provider establishes a direct TCP connection with all the buyers mentioned in the desired list. This off-chain channel is encrypted using transport layer security (TLS). Both parties proceed with the negotiation stage. During the negotiation, the provider requests bids from all the selected buyers who send their bids in return. Provider evaluates all these bids and can either reject and request a new bid or accept the bid or counter the proposed new bid. Once both parties accept a bid, a data agreement leveraging a smart contract is established between them. Both parties agree on the granularity of payment, which is the number of data transactions after which payment must be made. It is represented by the variable N. The benefit of using N is that if any of the involved parties perform any malicious activity, they can lodge the dispute early instead of waiting till the end of the contract. To prevent the malicious behaviour of any participant, both involved parties deposit an amount greater than the data price in their escrow account, which is locked till the subscription settlement happens. This is to account for the dispute resolution fee of a facilitator. If a participant is involved in any malicious activity, this deposit is used to penalize him. Provider shares a symmetric session key with the buyer to encrypt all the data in transit. Data is transmitted to the buyer as per the agreed specifications. The buyer verifies the received data and, in return, sends an acknowledgement to the provider. Buyer and provider use a metering system [31] to count N transactions. At the end of this, both parties initiate settlement and disconnect.

Some of these interactions are governed and automated by smart contracts. These contracts record the agreement details, participants' reputation scores, and data price in a decentralized, transparent, secure, and efficient manner in the blockchain for future use. The methods provided by each contract and the transaction details to execute them are given in detail below:

3.4.1 Data Subscription Contract (*SubscriptionSc*)

All the agreed and negotiated terms are encoded, compiled, and deployed in the blockchain as a smart contract known as the *SubscriptionSc*. For each provider-buyer pair, an instance of *SubscriptionSc* is spawned. Its state corresponds to the subscription details, such as *Device id, Data type, Start time, Interval, Duration, Subscription Status, Payment Granularity, Quality Score, Risk Score, Total cost and Negotiation Terms. Device id* is the identifier of the IoT device that generates data. *Start time* is the time when the subscription starts. *Interval* is the time between two data samples, and *Duration* is the data collection period. *Subscription Status* manages the execution flow of the subscription. Its value is assigned from a predefined set INITIATE, ACTIVE, SETTLEMENT, DISPUTE, FINISH, INSUFFICIENT FUNDS. Payment Granularity refers to the option to make a settlement into a smaller equal part. Quality Score quantifies the quality preference of the buyer, and Risk Score quantifies the data sensitivity per the provider's preference. Total cost is the amount of money needed to pay for the subscription. It comprises three components: Execution cost, Data cost and Facilitator fee. Execution of a smart contract incurs a fee referred to as the Execution cost. It is divided between the buyer and provider as per the negotiation terms. Data cost is the agreed-upon price of a data sample. It is used to determine the base price of a data type, as explained later in section 3.4.3. The facilitator fee is the payment in return for their service. *SubscriptionSc* uses *subscriptionAdd, subscriptionInfo, subscriptionStart, subscriptionSettlement and subscriptionDelete* methods. These are executed by transactions Tx_{addS}, Tx_{infoS}, Tx_{startS}, $Tx_{confirmDelivery}$, $Tx_{resolveS}$ and Tx_{delS} respectively to automate various stages of subscription as explained below:

- *subscriptionAdd()*: This method adds a new subscription to the subscription list. Only the provider can issue the transaction Tx_{addS} to prevent the buyer from maliciously adding any subscription to the contract. The subscription status of the newly added subscription is set to INITIATE. This emits an event *subscriptionAdded* to the provider and buyer with the subscription id (SID). Tx_{addS} is given as

$$Tx_{addS} = [\text{subscription detail}|Sig_p|PU_p] \tag{3.5}$$

  where Sig_p and PU_p are the signature and public key identifier of a provider, subscription detail consists of data specifications including device id, data type, start time, interval, duration, subscription status, payment granularity, quality score, risk score, total cost and negotiation terms.

- *subscriptionInfo()*: This function receives the subscription id and returns the corresponding subscription details. The read-only transaction Tx_{infoS} can be issued by either of the concerned parties where SID is the subscription id, Sig and PU are the signature and public key of either the provider or buyer.

$$Tx_{infoS} = [SID|Sig|PU] \tag{3.6}$$

- *subscriptionStart()*: After the buyer verifies the subscription details using the received SID, they starts the subscription by issuing Tx_{startS}. Moreover, at the subscription start time, the provider also issues Tx_{startS}. On receiving transactions from both parties, this method changes the subscription status of SID to ACTIVE.

$$Tx_{startS} = [SID|Sig|PU] \tag{3.7}$$

- *subscriptionSettlement()*: This method is executed by $Tx_{confirmDelivery}$ where D_{count} is a non-zero integer while feedback F is a real number in $[0, 1]$.

$$Tx_{confirmDelivery} = [SID|D_{count}|F|Sig|PU] \tag{3.8}$$

At the end of the subscription, both actors submit their data count confirming the delivery of D_{count} data samples and provide feedback F to the other actor based on their experience. On receiving $Tx_{confirmDelivery}$ from both actors, the subscription status changes to SETTLEMENT. During settlement, data counts sent by both parties are compared, and the following actions are taken.

Case 1: Data counts submitted by both actors match. This means no conflict occurred. Invoice and payment are released to the buyer and provider, respectively, marking the subscription status to FINISH.

Case 2: When data counts mismatch, the actor with the higher reputation score is trusted, and payment settlement is performed based on their claimed data count. In this case, the subscription status is set to DISPUTE. However, to prevent an actor from misusing their reputation score, an affected actor can report the dispute to the facilitator at the end of payment granularity. The affected actor can raise the complaint by sharing AID and SID of the disputed subscription with the facilitator. A facilitator identifies and penalizes the misbehaving actor by using the evidence stored in the blockchain and, in return, receives a fee. The facilitator issues $Tx_{resolveS}$ to submit their feedback for the actor (PU), hence, changing the subscription status from DISPUTE to FINISH.

$$Tx_{resolveS} = [PU|F|Sig_F|PU_F] \tag{3.9}$$

- *subscriptionDelete*(): A provider issues Tx_{delS} to delete the subscription entry from the subscription table. Only the entry with subscription status set to FINISH gets deleted.

$$Tx_{delS} = [SID|Sig_p|PU_p] \tag{3.10}$$

3.4.2 Register Contract (*RegisterSc*)

This is a multisig contract, which ensures that both parties have agreed upon the deployed contract terms and conditions. It keeps records of all the registered users using method *registerActor* executed by the transaction $Tx_{registerActor}$. Moreover, it also maintains a *SubscriptionSc* lookup table comprising *Contract ID, Provider identifier, Buyer identifier, SubscriptionSc address.* Various methods, including *registerAgreement, removeAgreement, getAgreement* are employed to manage the lookup table using transactions $Tx_{registerA}$, $Tx_{removeA}$ and Tx_{getA} respectively. In addition, *RegisterSc* also implements methods including *setPrice(), getPrice(), setRepS(), getRepS()* to interact with *PriceSc* and *ReputationSc* respectively.

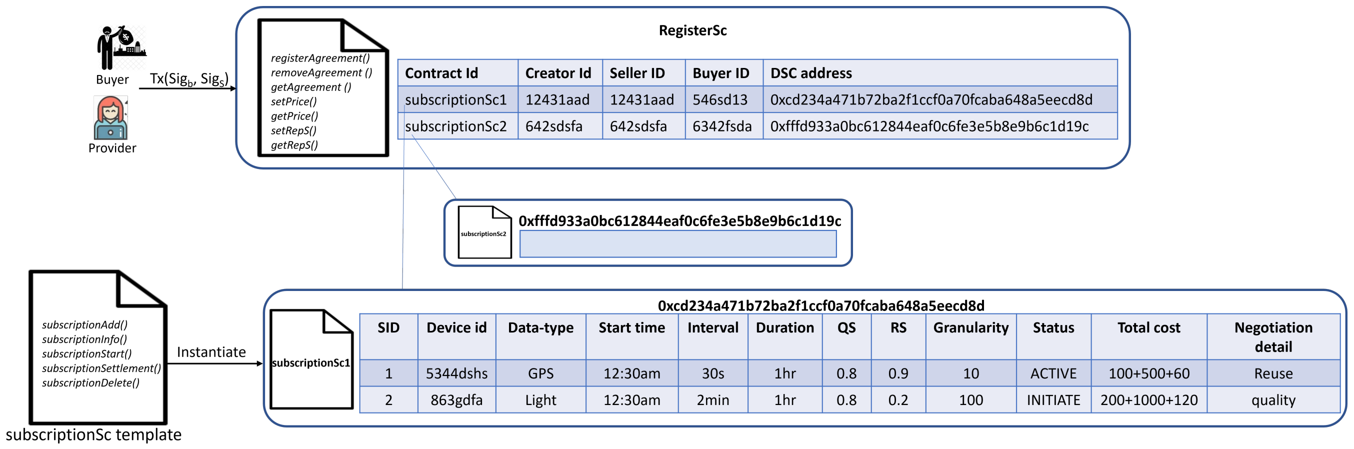
- *registerUser*(): A new actor registers in the marketplace by issuing transaction $Tx_{registerUser}$ given by Equation 3.11. *Role* can either be provider or buyer, *Sig* and *PU* are the signature and public key of the actor.

$$Tx_{registerUser} = [Role|Sig|PU] \tag{3.11}$$

- *registerAgreement*(): A newly deployed *SubscriptionSc* can be added to the contract lookup table using multisig transaction, $Tx_{registerA}$, where *address* is the address of the deployed *SubscriptionSc*. Before creating a new entry of *SubscriptionSc*, this method verifies that both parties send the registration request. This ensures that both parties agree with the terms mentioned in the *SubscriptionSc*. It emits the event *agreementCreated*, returning the agreement ID AID.

$$Tx_{registerA} = [address|Sig|PU] \tag{3.12}$$

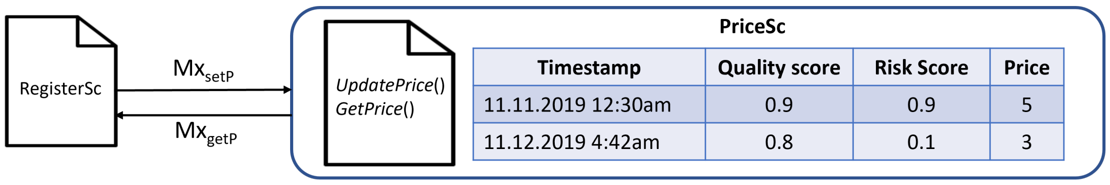
- *removeAgreement*(): This method deletes the contract entry from the lookup table. It subsequently performs the *SelfDestruct* operation of the *SubscriptionSc* contract



Contract Id	Creator Id	Seller ID	Buyer ID	DSC address
subscriptionSc1	12431aad	12431aad	546sd13	0xcd234a471b72ba2f1ccf0a70fcaba648a5eecd8d
subscriptionSc2	642sdsfa	642sdsfa	6342fsda	0xfffd933a0bc612844eaf0c6fe3e5b8e9b6c1d19c

SID	Device id	Data-type	Start time	Interval	Duration	QS	RS	Granularity	Status	Total cost	Negotiation detail
1	5344dshs	GPS	12:30am	30s	1hr	0.8	0.9	10	ACTIVE	100+500+60	Reuse
2	863gdfa	Light	12:30am	2min	1hr	0.8	0.2	100	INITIATE	200+1000+120	quality

Figure 3.7: Functions and data structures of *SubscriptionSc* and *RegisterSc* contracts



Timestamp	Quality score	Risk Score	Price
11.11.2019 12:30am	0.9	0.9	5
11.12.2019 4:42am	0.8	0.1	3

Figure 3.8: Functions and data structures of *PriceSc* smart contract

to remove the code and storage of the *SubscriptionSc* from the blockchain, such that the *SubscriptionSc* can no longer be available. It is invoked by the multisig $Tx_{removeA}$.

$$Tx_{removeA} = [AID|Sig|PU] \tag{3.13}$$

- *getAgreement*(): Tx_{getA} is issued by either of the parties to get the contract details, such as its address, co-parties details etc.

$$Tx_{getA} = [AID|Sig|PU] \tag{3.14}$$

Fig. 3.7 depicts the functions and data structures of *SubscriptionSc* and *RegisterSc* smart contracts.

3.4.3 Price contract (*PriceSc*)

A *PriceSc*, as shown in Fig. 3.8, is deployed for each data type traded in the marketplace. The *PriceSc* maintains a ledger with the following fields: *time stamp, data sample cost, quality score, and risk score.* These stored values are used to calculate the base price offered by other providers in the market. *PriceSc* provides two methods *updatePrice()* and *getPrice()* to update and retrieve prices. These are executed by transactions Mx_{setP} and Mx_{getP}. When a new subscription is added in the *SubscriptionSc*, it triggers *RegisterSc* to issue Mx_{setP} to the *PriceSc*. Mx_{setP} records the latest agreed-upon price of the data sample in the ledger. Data sample cost is derived from the data cost and the number of samples. Mx_{getP} retrieves the base price of a data type *d*. These transactions are given by:

$$Mx_{setP} = [t|Price^{d}(t)|QS|RS] \tag{3.15}$$

$$Mx_{getP} = [ID_{d}|t_{base}] \tag{3.16}$$

where t is the time when the new subscription is added, $Price^{d}(t)$ is the agreed upon per data sample price given in the subscription, QS is the quality score, and RS is the risk score. ID_d is the identifier of data type *d*. These factors are used to determine the base price, as explained below.

We have adopted competition-based pricing in our framework because a data marketplace is highly competitive, with many providers interested in offering their IoT data in return for incentives. In competition-based pricing, the provider can use competitors' prices for the same data type as the basis for setting a price. This strategy does not require complex computations [148, 149] and varies depending on the market. *RegisterSc* deploys a new *PriceSc* when a new data type is traded. A provider can select the cost of new data type per the market dynamics of supply-demand [147]. Then providers can choose to follow the market going rate of a certain data type as charged by other providers selling the same data type. Nonetheless, a provider may provide value-added features besides price to draw the buyer's attention. These value-added features could be the data satisfying the high-quality demand of buyers or posing a high risk of privacy violation for the provider. We quantify these value-added features by (i) Quality Score (QS), which is the weighted average of the buyer's demand for quality levels and preferences, and (ii) Risk Score (RS), which is calculated using the risk matrix technique [163]. The price per sample of the data type d at time t is computed based on the quality score $QS(t)$, the privacy risk score $RS(t)$ and the base price $Price^{d}_{base}(t)$ given by Equation 3.17.

$$Price^{d}(t) = (1 + QS(t) + RS(t))Price^{d}_{base}(t) \tag{3.17}$$

These parameters are explained as given below:

- Base price: The competitor's price is used as a benchmark and corresponds to the index price of data type d. It is calculated using agreed-upon data sample price ($Price^{d}_{agreed}(t)$), $QS(t)$ and $RS(t)$, which are stored in the ledger. Equation 3.18 gives the price index for a subscription of data type d at time t.

  $$Price^{d}_{index}(t) = \frac{Price^{d}_{agreed}(t)}{1 + QS(t) + RS(t)} \tag{3.18}$$

  To calculate the base price, the price index is averaged for all the successful subscriptions in a time interval. For $N_{sub}(t_{base})$ number of subscriptions in a time interval t_{base}, base price ($Price^{d}_{base}(t_{base})$) of data type d is calculated by Equation 3.19.

  $$Price^{d}_{base}(t_{base}) = \frac{\sum_{t \in t_{base}} Price^{d}_{index}(t)}{N_{sub}} \tag{3.19}$$

- Quality score (QS): Quality score is evaluated based on the buyer's desired data quality. Data quality can be defined in terms of accuracy, conciseness, completeness, timeliness, etc. [164]. We calculate QS as the weighted average of the desired quality level ($q_1, q_2, ...q_n$) for n quality attributes and given by Equation 3.20.

  $$\text{QS} = w_1 * q_1 + w_2 * q_2 + ... + w_n * q_n \tag{3.20}$$

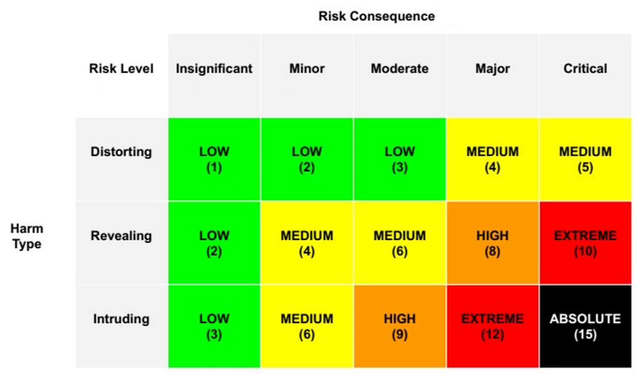
  where $w_1, w2, ...w_n$ are the weights representing the quality preference for each attribute.

- Risk score (*RS*): The risk score is determined using the risk matrix technique [163]. The IoT data holds intrinsic value, and sharing it exposes the provider to potential damage represented as risk, which can lead to several privacy harms. Therefore, a provider can categorize each data type into different harm types and the impact of risk represented as consequences as shown in Fig. 3.9a. Harm types depend on whether the privacy harm of the data type is distorting the provider's identity or reputation, revealing information about the provider or can potentially be used to expose or intrude into the provider's life. Consequences depend on the impact of the risk, ranging from insignificant (1) to critical (5). Based upon harm type and consequences preferred by the provider, a data type's risk level is estimated and is finally used to calculate its risk score, as shown in Fig. 3.9b.

3.4.4 Reputation contract (***ReputationSc***)

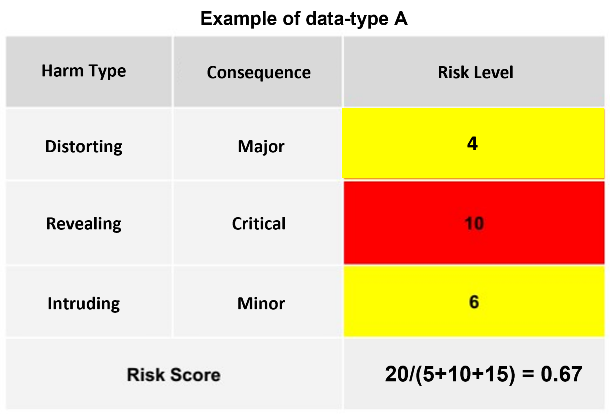
Reputation plays a key role in ensuring trustworthiness among actors who do not necessarily trust each other. MartChain employs a feedback-based reputation mechanism that considers the actor's past trading interaction and experience with other actors. Maintaining an actor's reputation score, represented as *RepS*, is essential in facilitating smooth transactions, building trust and reducing risk. We assume that $RepS \in [0, 1]$, such that the higher the value of *RepS*, the better the actor's reputation. The initial *RepS* of a new actor, whose history is unknown, is fixed at 0.5. A dishonest actor with low *RepS* can adopt a new identity to whitewash their reputation. Since an actor requires to register using $Tx_{registerActor}$ that consumes gas and incurs a fee, it will discourage whitewashing attack [165].

Our reputation model considers the actor's history that captures the number of transactions done and the value of feedback obtained by him. However, an actor with a good reputation might occasionally cheat, relying on the fact that a small number of negative feedback does not corrupt their reputation. To prevent such attempts, our model decreases the reputation value much faster for negative feedback than positive feedback increases it.



		Risk Consequence				
	Risk Level	Insignificant	Minor	Moderate	Major	Critical
Harm Type	Distorting	LOW (1)	LOW (2)	LOW (3)	MEDIUM (4)	MEDIUM (5)
	Revealing	LOW (2)	MEDIUM (4)	MEDIUM (6)	HIGH (8)	EXTREME (10)
	Intruding	LOW (3)	MEDIUM (6)	HIGH (9)	EXTREME (12)	ABSOLUTE (15)

(a)



Example of data-type A

Harm Type	Consequence	Risk Level
Distorting	Major	4
Revealing	Critical	10
Intruding	Minor	6
Risk Score		20/(5+10+15) = 0.67

(b)

Figure 3.9: (a) Risk Matrix (b) Risk Score Example

Nevertheless, since economic interests are relevant in the marketplace, a malicious actor could do the following actions or attacks in order to gain a reputation or to cheat by disturbing the trading of honest actors:

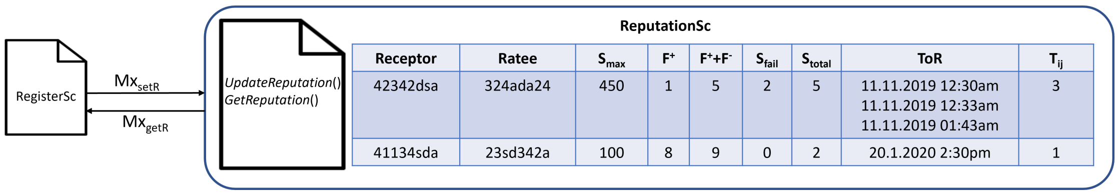
- Malicious reputation manipulation: An actor can exploit their *RepS* to malign other actor's *RepS* by giving false feedback.
- Strike and recharge attack: An actor may build up a good reputation by properly executing several low-value transactions or subscriptions and then misbehaving or cheating in high ones.

- Collusion attack: Two actors can collude to increase each other's *RepS*.
- Violation attack: An actor violates the agreement by disrupting the subscriptions. A malicious actor could start trading and then aborts it. Such behaviour causes damage to honest actors who spend resources, like the battery of the IoT device, which is crucial in IoT environments.

To overcome the above attacks, our reputation model determines the new reputation value by tuning the previous reputation value and the latest feedback. The tuning factor depends on various parameters, including feedback credibility (FC), subscription value (TV), collusive activity (CA) or violation factor (VF), as explained later. In order to show how the *RepS* is computed and updated, we assume that $Actor_i$ (a.k.a ratee) submits their feedback for $Actor_j$ (a.k.a receptor) using $Tx_{confirmDelivery}$. This triggers a message Mx_{setR} (Equation 3.21) to the *ReputationSc* to update their trade experience.

$$Mx_{setR} = [PU_j|S_{value}|F_{ij}|Sig_i|PU_i] \tag{3.21}$$

where PU_j is the identifier of $Actor_j$. Sig_i and PU_i are the signature and public key of $Actor_i$. S_{value} is the monetary value of the subscription. F_{ij} is the feedback provided by $Actor_i$ to $Actor_j$ that represents their experience about the transaction. We assume that $F_{ij} \in [0, 1]$, where $F_{ij} = 1$ increases with the level of satisfaction. *ReputationSc* maintains the feedback record list of each actor as illustrated in Fig. 3.10: maximum transaction value in the trade history (S_{max}) with an $Actor_i$, total number of negative feedback (F_{ij}^{-}) given by $Actor_i$ to $Actor_j$, the total number of feedback ($F_{ij}^{-}+F_{ij}^{+}$) given by $Actor_i$ to $Actor_j$, the total number of failed subscriptions S_{fail} due to the agreement violation, the total number of subscriptions S_{total}, list of all requests timestamp (ToR) and the total number of trade transactions $T_{i,j}$ between $Actor_i$ and $Actor_j$ within a given interval. *ReputationSc* provides two methods: *updateReputation()* to update the feedback in the list and *getReputation()* to retrieve the *RepS*. As a consequence of Mx_{setR}, the $RepS_j$ of $Actor_j$ is updated as follows:



Receptor	Ratee	S_{max}	F^+	F^++F^-	S_{fail}	S_{total}	ToR	T_{ij}
42342dsa	324ada24	450	1	5	2	5	11.11.2019 12:30am 11.11.2019 12:33am 11.11.2019 01:43am	3
41134sda	23sd342a	100	8	9	0	2	20.1.2020 2:30pm	1

Figure 3.10: Functions and data structures of *ReputationSc* smart contract

$$RepS_j = \begin{cases} ((1-\alpha)\overline{RepS_j} + \alpha F_{ij})e^{-\frac{S_{fail}}{S_{total}}} & \text{for successful subscription} \\ \overline{RepS_j}.VF & \text{for failed subscription} \end{cases} \tag{3.22}$$

where

$$\alpha = \frac{(FC+TV).CA}{2} \tag{3.23}$$

In the Equation 3.22, the new $RepS_j$ of $Actor_j$ is computed for successful and failed subscriptions by updating the previous reputation value ($\overline{RepS_j}$) of $Actor_j$. For successful subscriptions, the previous reputation value of $Actor_j$ is computed by tuning the feedback given by $Actor_i$. The exponential factor is the ageing factor of the impact of failed subscriptions with respect to the total subscriptions. The tuning factor, α, considers FC, TV and CA as given by Equation 3.23. While for the failed subscriptions, the violation factor (VF) is used to tune the previous reputation score. These parameters are explained below.

- *Feedback credibility* (FC): This factor denotes the credibility of the feedback source to prevent reputation manipulation attack and is calculated using Equation 3.24 as given in [166]. FC is high whenever (1) the reputation $RepS_i$ of $Actor_i$ providing the feedback is high and (2) the number of negative feedbacks (F_{ij}^-) provided by $Actor_i$ is small with respect to the total number of feedbacks ($F_{ij}^- + F_{ij}^+$).

$$FC = \frac{RepS_i}{RepS_i + RepS_j}.(1 - \frac{F_{ij}^-}{F_{ij}^- + F_{ij}^+}) \tag{3.24}$$

- *Subscription value* (TV): The use of this parameter prevents strike and recharge attack [166] by modulating the feedback with the transaction value. In other words, for a low-valued transaction, the positive feedback increases the reputation score slightly, but the increase is considerable for a high-valued transaction. TV is the ratio of the current subscription value (S_{value}) to the maximum value (S_{max}) of all the subscriptions happened between $Actor_i$ and $Actor_j$ as given in Equation 3.25.

$$TV = \frac{S_{value}}{S_{max}} \tag{3.25}$$

- *Collusive activity* (CA): A collusion factor [166] is employed to prevent the possibility of actors colluding with one another to increase or decrease each other's reputation scores by providing a number of (positive or negative) feedback. CA is negatively correlated to the number of previous transactions $T_{i,j}$ made between $Actor_i$ and $Actor_j$. CA is calculated using Equation 3.26 where a is a non-negative constant.

$$CA = (\frac{1}{T_{i,j}})^a \tag{3.26}$$

- *Violation factor* (VF): An agreement can be violated due to the provider's resource-constrained IoT devices or the buyer's malicious intention to disrupt the data service. The violation factor (VF) defines the reliability of the actors in such cases. The reputation score of the actors drastically decreases if they fail to comply with the agreed subscription. This is to ensure that the provider delivers the data to which they agreed and that the buyer does not cheat in honest transactions. VF is calculated using the number of failed subscriptions (S_{fail}) and the total number of subscriptions (S_{total}) as given in Equation 3.27.

$$VF = 2 - 2^{(\frac{S_{fail}}{S_{total}})} \tag{3.27}$$

3.4.5 Autonomous data trading using smart contracts

Fig. 3.11 depicts how different contracts, as discussed in section 3.4.1-3.4.4 interact with each other to automate the execution flow of the subscription for data trading purposes. In Step 1a, a provider and a buyer reach an agreement via data trading protocol. In Step 1b, a new *SubscriptionSc* is compiled and deployed on the blockchain. In Step 2a, the provider registers the contract address of the deployed *SubscriptionSc* to *RegisterSc*

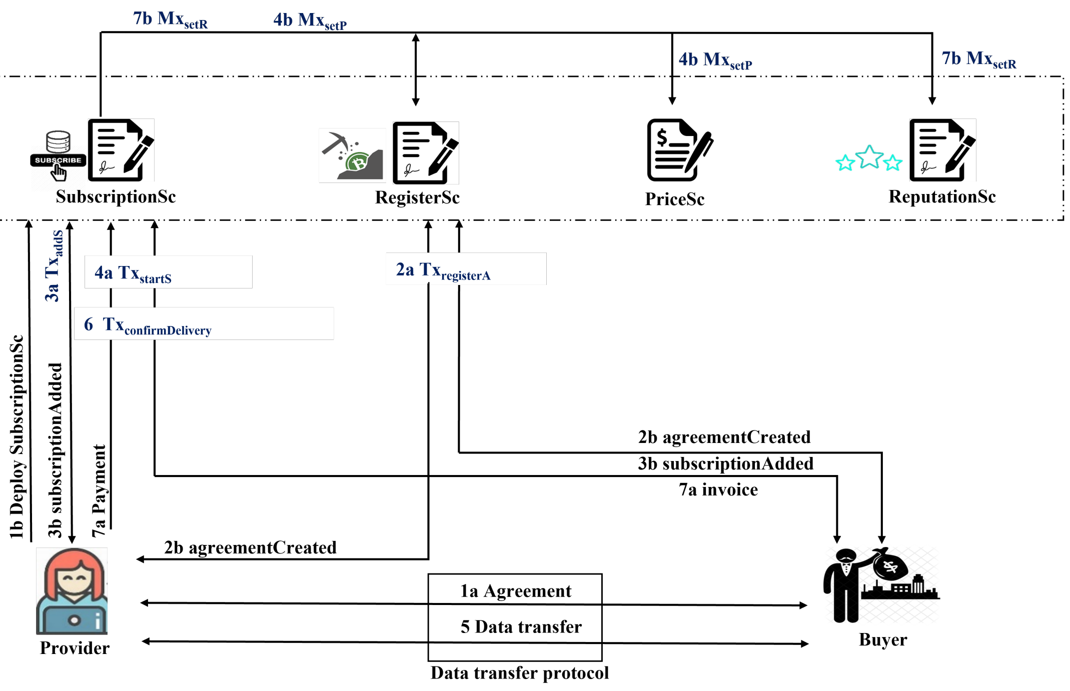


Figure 3.11: Autonomous data trading using smart contracts

using $Tx_{registerA}$. In Step 2b, *RegisterSc* emits the event *agreementCreated* returning the contract ID AID. Step 3a: Provider issues Tx_{addS} to add subscription details based on their negotiated terms. The subscription status is set to INITIATE. Step 3b: An event *subscriptionAdded* is emitted to the buyer with SID, who then verifies the subscription specification. Step 4a: If everything looks fine, the buyer issues Tx_{startS}. An equivalent amount corresponding to the total cost is held in their escrow account. If the buyer does not have the corresponding balance in their account, the subscription status is set to INSUFFICIENT FUNDS. At the subscription start time, the provider also issues Tx_{startS} that, in turn, changes the subscription status to ACTIVE. Step 4b: At this point, Mx_{setP} updates the price of the data type in the corresponding *PriceSc*. Then, the provider and the buyer engage in data transfer via data trading protocol in Step 5. Both parties issue $Tx_{confirmDelivery}$, and based on the count information sent by them *SubscriptionSc* verifies whether there is a conflict or not in Step 6. If there is no dispute, in Step 7a, *SubscriptionSc* issues an invoice to the buyer and payment is made to the provider. This marks the subscription status to be FINISH. If the data count does not match, then the DISPUTE is lodged, and payment is processed as per the dispute resolution explained earlier. In Step 7b, The entities also send feedback using $Tx_{confirmDelivery}$. *SubscriptionSc* issues Mx_{setR} to the *RegisterSc*, which in turn forwards it to the *ReputationSc*. It uses respective feedback to update the buyer's and provider's reputation scores.

3.5 Implementation and Evaluation

In this section, we provide a proof-of-concept implementation of the smart contracts, discussed in section 3.4, using the Solidity version 0.5.12. We thoroughly tested a single buyer and provider on the private Ethereum blockchain Ganache. This ensures that the logical flow and execution outcome from the interactions of different smart contracts are as expected. Then we analyze the cost of trade on an Ethereum public testnet. Next, we implement the NR-based algorithm, presented in section 3.3, in MATLAB to solve the EDSA problem. The simulation results provide insights into the provider's revenue generation in different scenarios. Finally, we provide a high-level comparative analysis of the existing blockchain-based approaches for the data marketplace as discussed in Chapter 2 with MartChain.

3.5.1 Proof of concept implementation

The implementation was done on the Ropsten testnet network, the public Ethereum network, which behaves similarly to a production blockchain. It runs a proof-of-work consensus and is used for testing purposes. Our smart contracts[1] were implemented and deployed using Remix IDE[2].

It is worth noting that any operation or transaction that modifies the state incurs fees, which need to be paid by the involved parties. These costs are estimated using the amount

[1]Smart contract codes available at https://github.com/pooja239/DataMart

[2]http://remix.ethereum.org/

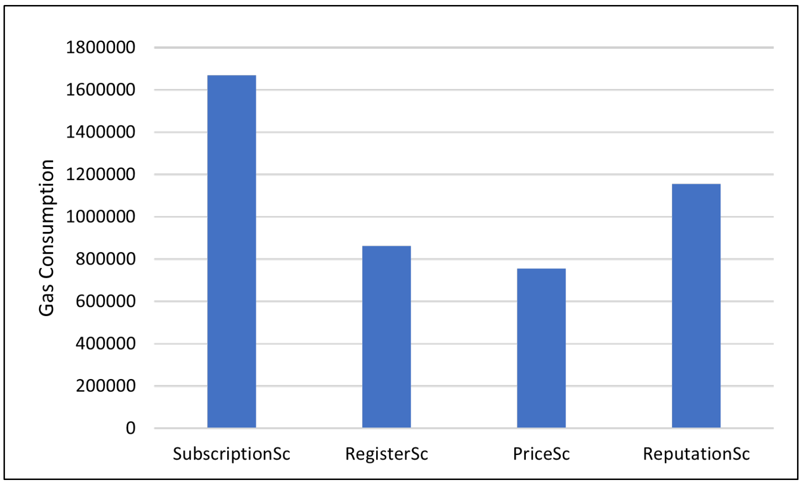


Figure 3.12: Deployment Cost

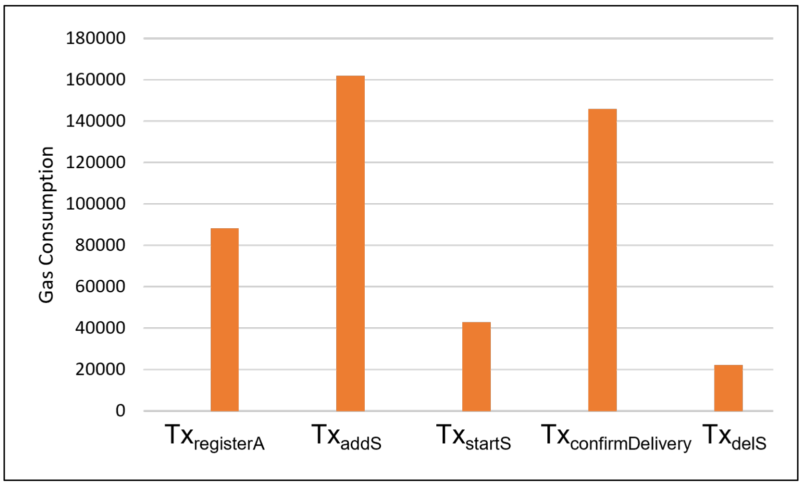


Figure 3.13: Tx Execution Cost

of gas consumed and the unit gas price. The gas consumed during any operation reflects the computational complexity or size of the smart contracts, while the miners in the system determine the gas prices. We used 10 Gwei gas price to evaluate the cost of different operations during data trading (the recommended gas prices can be found at ["ETH Gas Station", Ethgasstation.info, 2020. [Online]. Available: https://ethgasstation.info/. [Accessed: 23- Jan- 2020]).

Fig. 3.12 shows the deployment cost of contracts. It can be observed that *SubscriptionSc* and *ReputationSc* contracts consume the maximum amount of gas because of the

Table 3.3: Evaluation parameters for simulating EDSA

Parameter	**Value**
Number of IoT devices (i) per provider	$D = 5$
Data type (j) for each device	$S = 5$
Price of j^{th} type on i^{th} device	$P_{ij} \sim \mathcal{N}(10,\, 3)$
Quality of j^{th} type on i^{th} device	$Q_{ij} \sim \mathcal{U}\{10, 20, ..., 100\}$
Battery of i^{th} device (mAh)	$B_i = 2000$
Number of buyers (k)	$C = 10$
Total number of demands	$dem = 50$
Duration (hr) of k^{th} buyer demand for j^{th} type	$d_j^k \sim \mathcal{N}(5,\, 2)$
Sampling interval (min) of k^{th} buyer demand for j^{th} type	$s_j^k \sim \mathcal{N}(10,\, 60)$
Quality demanded by k^{th} buyer	$q^k \sim \mathcal{U}\{10, 20, ..., 100\}$

complexity of the functions involved in reputation score evaluation and maintenance of subscription execution flow. As discussed in section 3.4, end-to-end data trading involve the following transactions: *SubscriptionSc* contract deployment, $Tx_{regitserA}$ to register the *SubscriptionSc* contract address in the lookup table, Tx_{addS} to add the subscription detail, Tx_{startS} to start the subscription, $Tx_{confirmDelivery}$ to begin the settlement process followed by the Tx_{delS} to delete the completed subscription from the blockchain. The gas consumption of these transactions is shown in Fig. 3.13, where the total cost of all the operations is USD 2.67555. It can be observed that Tx_{addS} and $Tx_{confirmDelivery}$ consumed more gas than other transactions. This is because these transactions involve write operations on the blockchain. Tx_{addS} requires interaction with the *PriceSc* to fetch the price of the subscription and also update the pricing information in the ledger. At the same time, $Tx_{confirmDelivery}$ interacts with the *ReputationSc* to calculate and update the reputation score based on the latest feedback.

3.5.2 Evaluation

Table 3.3 gives the values of the parameters defined in section 3.3, used in the simulations unless stated otherwise. The performance results are generated by averaging results over 1000 iterations.

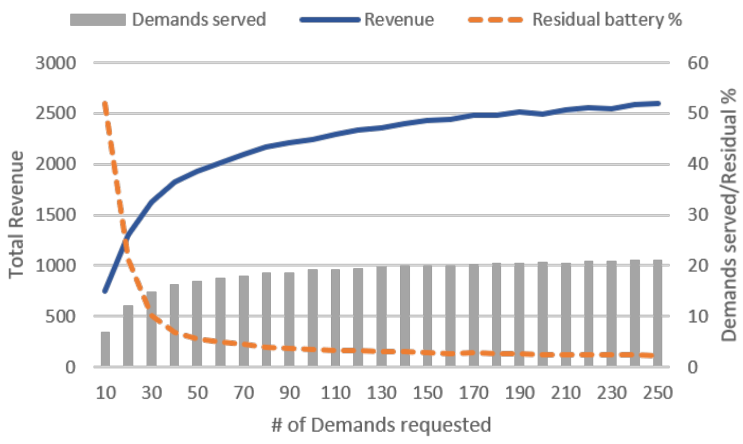


Figure 3.14: Revenue generation and demand selection with NR-based algorithm

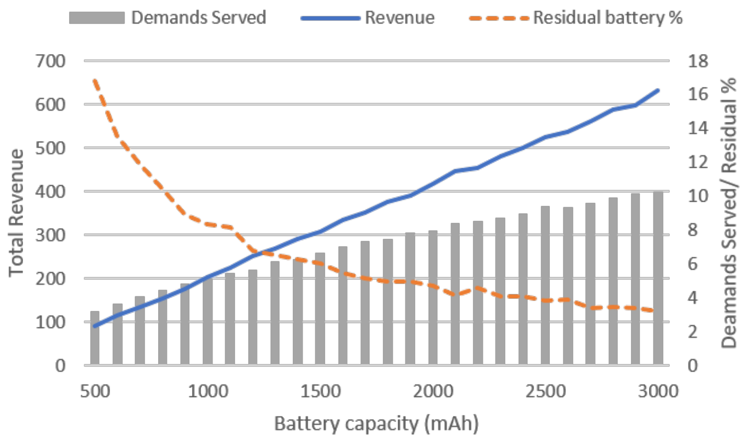


Figure 3.15: Effect of increasing battery capacity

Fig. 3.14 shows the effect of increasing the number of demands requested for 10 data types by 50 buyers. We can observe that increasing the number of demands requested from 10 to 250 causes a logarithmic growth for the revenue, which reaches saturation at a certain demand level (210-230) due to the limited batteries of the devices. This can be observed from the residual battery levels. Note that buyers' quality requirements may also limit the number of demands served by the provider, as some demands may have higher quality requirements than the quality of data provided by the provider.

Next, Fig. 3.15 demonstrates the effect of increasing the battery capacity on a provider's

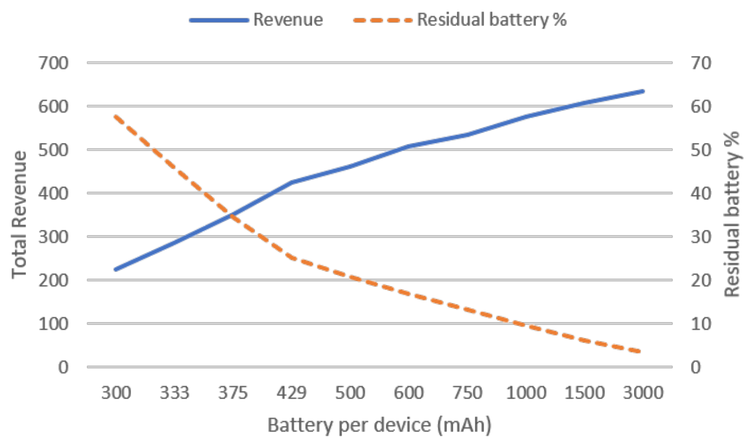


Figure 3.16: Effect of the increasing number of devices

total revenue with a single device. We observe that the revenue increases linearly with the increase in battery capacity from 500mAh to 3000mAh. As the battery capacity increases, the provider can select and serve more demands and generates higher revenue. The residual battery does not change much; however, the percentage of the residual battery decreases, resulting in better battery capacity utility. It is observed that the total revenue generated increases linearly with the linear increment of battery capacity.

Finally, we analyze the impact of increasing the number of provider devices on the generated revenue in Fig. 3.16. The overall battery capacity with a provider is kept fixed to capture the original trend of increasing the number of devices rather than increasing battery, which will linearly increase the generated revenue. We observe that increasing the battery capacity for IoT devices is more strongly related to revenue generation than increasing the number of devices. As we move from 10 devices with 300mAh batteries to 1 device with 3000mAh battery, the total revenue increases. This could be explained by the smaller battery capacities and residual batteries remaining on the devices. Devices with small batteries cannot serve demands that require high energy consumption. Furthermore, the residual batteries remaining on the devices cannot be combined and utilized to serve more demands.

3.5.3 Comparative Analysis

In this section, we present the comparison of existing blockchain-based marketplace approaches with our proposed approach across multiple dimensions, including usability, advantages, and disadvantages. As discussed in Chapter 2, these existing approaches include trade transaction management, distributed data catalogues, hybrid centralized-decentralized architectures, decentralized data storage access control, agreement instantiation. This analysis is useful to observe the potential benefits and drawbacks of using the MartChain platform for building decentralized data marketplaces in trading data in real-time from resource-constrained IoT devices. Table 3.4 summarizes the comparative analysis. It can be noted from Table 3.4 that existing approaches have advantages such as auditing and transparency, reliability in listing providers' offerings, high feasibility, enabling owner-controlled data, and non-repudiation respectively. However, these approaches also have some disadvantages, such as a lack of scalability, limited service offerings, centralization issues, high redundancy, and a focus on specific marketplace functionality respectively. In contrast, MartChain offers autonomy through the use of smart contracts to manage subscription workflows automatically, without any intermediaries. It also enables energy-efficient selection and allocation of buyer's demands on the resource-constrained IoT devices of the provider. Besides, MartChain enforces a holistic model equipped with all essential components of the marketplace, such as discovery and selection, data agreement management, price model and reputation mechanism (see chapter 2). Recall from section 3.2, MartChain requires decentralized database systems such as BigChainDB or blockchain-based data catalogue approach in tier-1 to maintain the device and data information transparently and efficiently. Overall, the comparison suggests that MartChain offers a promising approach for autonomous and efficient real-time data trading, but further research is needed to optimize its performance and scalability.

Table 3.4: A comparative analysis of the existing blockchain-based approaches in the data marketplace with MartChain

Approach	**Usability**	**Advantage**	**Disadvantage**
Trade transaction management	Stores trade logs	Auditing and transparency	Lack of scalability
Distributed data catalogues	Records data offerings	Reliability in listing provider's offering	Limited offered services
Hybrid centralized-decentralized architectures	Pays the price involved in the trade	High feasibility	Centralization issues
Decentralized data storage Access control	Maintains access policy	Owner controlled data	High redundancy
Agreement instantiation	Manages agreement details	Non-repudiation	Specific functionality of marketplace
MartChain	Records subscription details, data pricing trend, participant's behaviour	Autonomous, efficient, full-fledged with key marketplace functionalities	Requires two separate blockchains

3.6 Chapter Summary

MartChain is a fully-functional and effective decentralized marketplace with essential components such as discovery and selection, data agreement management, price model and reputation mechanism. We outlined the participant's roles in 2-tier architecture and interaction among them. We presented functionalities of four smart contracts (*SubscriptionSc*, *RegisterSc*, *PriceSc*, and *ReputationSc*) and their associated methods to achieve the trustful automated data trading by eliminating the third-party risk. The framework also provides a simple and dynamic way of pricing the data based on the competitors' prices in the market and encourages providers to sell higher-quality data and value-added features in return for greater incentives. In order to provide resilience to various security attacks, a reputation mechanism is presented that uses the trading history of entities in tuning the reputation score and penalizes them for being dishonest. The MartChain is underpinned by an EDSA mechanism for optimally selecting and allocating buyers' demands on provider's resource-constrained IoT devices while satisfying the battery, quality and allocation constraints. EDSA maximizes the provider's revenue while meeting the buyers' requirements and ensuring the completion of the selected demands without any interruptions. We presented preliminary work on the Ethereum blockchain showing how to implement these components using smart contracts. Also, we evaluate the gas consumption and cost incurred for interacting with the proposed marketplace framework. Furthermore, we formulated the EDSA problem in MATLAB and simulated various scenarios to analyze the impact of the requested demands on the battery drainage of the provider's devices. Our result shows that our approach is viable and benefits the provider and buyer by creating an autonomous and efficient real-time data trading model.

Chapter 4

KYBChain: Know-your-buyer in Privacy-aware decentralized IoT data marketplace

This chapter answers the research questions **RQ4**, **RQ5**, **RQ6**, and **IQ**. In a data marketplace, providers are driven mainly by their economic interests and may willingly disclose their valuable and sensitive IoT data to buyers. However, they are often unaware of the long-term privacy risks associated with trading or sharing their IoT data. Three characteristics of the buyer that raise privacy concerns and risks include (i) Non-compliance: a buyer may not have adopted or complied with the industry standard privacy practice and security measures to safeguard the provider's data; (ii) Data accumulation: Large accumulation of data by the buyer can make them a target of cybercrime or a buyer acting in bad faith can misuse this data to infer highly personal information about the provider; (iii) Data-leak: a buyer may intentionally or accidentally share the provider's sensitive data with an unauthorized party without the provider's knowledge or consent. Various technical measures, including privacy labels, privacy-enhancing techniques, data leakage prevention mechanisms, data provenance models, etc., have been developed to minimize or prevent such concerns and risks. A provider must trust the buyer to implement or follow these measures properly. Currently, providers rely on trust scores to measure the buyer's trustworthiness in managing their privacy. However, the trust score is a subjective metric that depends on providers' historical experiences and opinions. Therefore, to limit the role of subjectivity in the providers' decision to data-sharing with a buyer, it is essential to provide them with factual and relevant information to make them aware of the associated privacy risks. In this chapter, we approach the privacy issue from the provider's viewpoint and empower them to make privacy-aware data-sharing decisions. We propose a framework, KYBChain, to compute a privacy rating of the buyer that the provider can use to manage his risk. This rating is defined specifically for a buyer-provider pair, which measures the provider's overall and long-term risk caused by sharing his data with the buyer. Our definition of privacy rating (PR) satisfies the following intuitive properties: better privacy practices and security measures reduce the provider's privacy risks and increase his PR; higher the sensitivity and visibility of accumulated data greater the provider's privacy risks and lower his PR; higher the probability and severity of data leak risk, greater the provider's privacy risks and lower his PR. We identify several privacy attributes/elements to model and develop a methodology to formulate PR satisfying the above-stated properties. KYBChain employs blockchain and smart contracts to record all these privacy attributes/elements to evaluate PR in a transparent, efficient, secured and automated manner. We conduct several experiments on synthetic data to demonstrate the efficacy and practical utility of PR. Furthermore, we present the proof of concept implementation of KYBChain in a private Ethereum network and analyse the overheads to demonstrate its feasibility.

4.1 Introduction

The adoption of Blockchain technology in MartChain (see Chapter 3) establishes non-repudiation, trustworthiness and transparency among the participants. A provider receives short-term economic gain by trading their IoT data in such a marketplace. As their engagement increases, trading data with a particular buyer over the longer term may have repercussions on their privacy [163]. Moreover, data is considered an asset [167] and requires protection even after being sold. Therefore, it becomes the buyer's responsibility to safeguard the provider's data privacy. Privacy regulation compliance is important for holding or managing a provider's data. Buyers must adopt privacy practices and security measures to safeguard the provider's purchased data. Other liabilities arise from the risk that a large accumulation of data could make the holder a target for cybercrime. Moreover, a buyer acting in bad faith can misuse all the purchased data and infer highly personal information about providers resulting in various predictive privacy harms such as localization and tracking, profiling, etc. To mitigate this threat, the buyer must use the data as per the agreed purposes and delete it after its retention period. Nevertheless, even when they do so, a state of zero risk remains unachievable. Buyers that strive for compliance can still fall victim to data leaks. When a provider perceives the buyer is incapable of protecting their privacy, they associate risk with data sharing. They may decide not to engage or restrict their level of engagement with the buyer. Therefore, it becomes increasingly important that the provider can accurately assess the overall and long-term risk of sharing their data with a buyer. To fulfil this aim, an IoT data marketplace must provide measures that can help a provider (i) make a privacy-aware informed decision of data disclosure to a buyer and (ii) manage and minimize their associated long-term privacy risk. To design privacy-aware systems [85], understanding a provider's preferences and concerns is of great importance. Factors such as trade interactions, transparency of mechanisms, context and who is purchasing the data are essential determinants. Three sources of such privacy concerns that may arise from the buyer's characteristics based on their dynamic activities are non-compliance risk, data accumulation risk and data leakage risk, as explained in detail below:

Non-compliance risk: This refers to the risk that the buyer may use the data in a way that violates privacy regulations or other laws. For example, in 2018, Facebook had allowed third-party app developers to access users' data without their consent, which violated a 2011 FTC consent decree [19]. Currently, buyers are provided with various strategies to notify providers of their practices, such as publishing or providing written privacy policy statements. Privacy policies [168] are legal documents that detail the purpose of usage, user's control, retention period, security practices, etc. These comprehensive documents are designed to give confidence to the provider that the buyer will safeguard their privacy by complying with the stated privacy regulations and security practices. However, privacy policies are long, complex, and contain complicated legal jargon. Therefore, most providers typically ignore or skip reading them [169]. In recognition of the ineffectiveness of privacy policies, numerous research [98, 99] have addressed the problem of making privacy policies more understandable and simplified. Nevertheless, it is still challenging for a provider to determine if the buyer complies with the data protection practices and security measures stated in their privacy policy.

Data accumulation risk: This risk refers to the potential that a buyer may accumulate privacy-sensitive data over time from a provider. Large data accumulation may increase the risk of a single point of cyber attack or failure because of data centralization. Moreover, a buyer acting in bad faith can easily misuse this data and infer highly personal information about the provider [170]. This could be used for nefarious purposes such as behaviour profiling [171], data profiling [172], individual identification [173], tracking [174], identity theft, or targeted advertising. Hence, exposing the provider to undesirable consequences and privacy violations. For example, in 2013, Target, a major US retailer, used customer data to create a pregnancy prediction score for each customer, which was used to send targeted ads for baby products. However, this led to an incident where a teenager's pregnancy was discovered by her father due to Target's targeted ads, causing significant privacy concerns for the family [175]. Additionally, the privacy risk of the provider increases as their disclosed data becomes more visible in the marketplace. For instance, suppose a provider sold their IoT data to several buyers with consent to reshare it with other buyers in the marketplace. A deceitful data broker acting as a buyer can buy the provider's data directly from him and indirectly through other buyers. Using advanced analytics, the data broker can build up a detailed profile [176, 177] of the provider and sell it to others. In this regard, conventional studies [178] have mainly focused on privacy-enhancing techniques such as anonymization, de-identification and decentralized technologies. However, these techniques suffer from various limitations. Data anonymization [179] refers to an irreversible transformation of data to prevent identifying a particular individual. Many studies [180, 181] have shown that data could be combined with metadata that is available openly on the Internet can allow such re-identification. De-identification [182] is the process of removing personal information from a dataset. However, IoT data is often very difficult to de-identify due to its highly granular nature [183, 184]. Besides, longitudinal data, i.e., data collected over time, e.g. mobility traces, is difficult to de-identify, even when aggregated. Decentralized and distributed technologies, such as blockchain, enable individuals to control data sharing by removing any intermediaries and improve transparency [185,186] by recording all the interactions on the immutable ledger. With control and transparency, a provider can track all the data they has traded with others in the marketplace. However, it is still hard for a provider to quantify their privacy exposure, which may increase over time due to the sensitivity of data the buyer has accumulated and the visibility of disclosed data to others.

Leakage risk: This risk refers to the possibility that the buyer leaks sensitive or protected data of a provider to an unauthorized third party either intentionally (e.g., resold data for monetary benefits) or inadvertently (e.g., due to failure to implement adequate privacy or security measures). For example, in 2017, Equifax, one of the largest credit reporting agencies in the US, suffered a massive data breach that exposed the personal information of over 143 million people [187]. The breach occurred due to a vulnerability in Equifax's system, which allowed hackers to gain unauthorized access to the data. This breach included sensitive data such as Social Security numbers, birth dates, and addresses. Existing literature employs watermarking [137, 188] to detect intentional data leakage. Based on a similar approach, we proposed TrailChain (see Chapter 5) that uses watermarking to track the data ownership and detect legitimate and illegitimate data reselling within and across marketplace systems. However, a buyer can also intentionally leak the data outside the confines of the marketplaces due to malicious insiders [189]. Furthermore, to prevent accidental disclosure of data to unauthorized entities, a buyer can employ several security mechanisms [190] such as firewalls, virtual private networks, intrusion detection systems or data leakage detection or prevention systems (DLPD). Even though the purpose of DLPD systems [191] is to ensure that sensitive data is not misused or accessed by unauthorized users, a provider is often unaware of the privacy risks of sharing data with a buyer who was involved in a data leakage event in the past. Such an incident will make the provider sceptical about the buyer's ability to handle sensitive information, reducing their willingness to sell more data to him. Moreover, a buyer hoarding a large variety of sensitive data from providers will be at higher risk of cybercrime. While deciding to sell data to the buyer, the provider is often unaware of the privacy risks of data leakage or the impact of the historical breach events associated with the buyer.

It's important to note that the examples cited above are not specific to IoT devices, but they illustrate the risks associated with sharing personal data in general. The risks associated with IoT devices may be similar, but they may also have additional considerations due to the nature of the data collected and the ways in which it is used. Consequently, a buyer in a marketplace can appear as a black box to the provider. Provider's confidence that their data is handled appropriately by the buyer relies primarily on trust. A provider must trust that the buyer will: comply with privacy regulations and security best practices in their policy, not accumulate the data and misuse it to invade their privacy and exert constant effort to mitigate any unauthorized sharing with third parties. Current data marketplace ecosystems rely on trust management models [33, 60, 91] that evaluate the trust score of a buyer based on their interactions, historical behaviour, experience, feedback or ratings submitted by providers. Buyer's trust score governs the provider's decision of whether or not to give data access to them [192]. However, trust is a subjective belief about an entity in a particular context [92]. It lacks factual and relevant information about buyers that can empower providers to properly assess the aforementioned privacy risks and make a privacy-aware informed decision about data disclosure to them.

In this chapter, we address the privacy issue from the provider's perspective by providing him with relevant information about the buyer. To this end, we maintain three profiles for a buyer, namely practice, purchase and leakage, that capture their various characteristics regarding the three aforementioned concerns. The practice profile is based on assessing the buyer's privacy and security measures and keeps track of any updates in their data protection practices. The purchase profile is a provider-dependent characteristic and monitors all the data purchase transactions of the buyer specific to a provider. A leakage profile maintains the records to assess the buyer's data leak risk. However, even if providers have access to all these profiles, it will be difficult for them to digest the information within them [193]. Therefore, it is necessary to provide providers with an effective and simple way to ascertain the implications of selling data to the buyer on their privacy. On that account, one meaningful approach is to implement a standardized and simple indicator for each buyer-provider pair, known as privacy rating (PR). This new privacy visualization can estimate the privacy risk of data sharing with a buyer and limit the role of subjectivity in the provider's decision. We also define a privacy rating vector (PRV) for a buyer to enable fine-grained decision-making according to the providers' preferences for each profile. PRV comprises an overall rating (PR) and more specific ratings corresponding to the practice, purchase and leakage profile of a buyer represented as PR_{pra}, PR_{pur} and PR_{lea} respectively. Privacy rating can offer many benefits to providers, buyers, and regulators. Providers can act according to their privacy preferences by comparing PR or PRV of different buyers and make a privacy-aware data-sharing decision. The transparency resulting from the published assessments in PRV creates an incentive for buyers to include secure design and privacy practices to get a good rating. They will be able to confidently demonstrate their commitment to protecting the privacy of providers. Furthermore, regulators can use PRV to bring enforcement actions against buyers who fail to protect providers' data adequately.

Current approaches for determining privacy ratings are based on data collection in different contexts, such as mobile applications [194, 195], social networks [101, 102] or websites [103]. These approaches do not consider the privacy elements/attributes that impact the providers' aforementioned privacy concerns in a data marketplace. Furthermore, scores/ratings have been shown in other contexts such as food [196], energy [197, 198] or mobile application ratings [199] to significantly impact user's decisions by effectively communicating important information to them. However, these mechanisms use static details such as information present in the privacy policy for website [103] or IoT devices [100, 200, 201] or privacy settings in mobile applications [194, 195] or social networks [101, 102], etc. The evaluation of the buyer's privacy rating should be dynamic. Furthermore, all these existing rating systems rely on a TTP to manage and store all the rating data required to evaluate buyers' ratings. Since a TTP can collude with the buyer to change the rating in their favour by removing, modifying, and manipulating all their rating data, such systems lack trust in data integrity and are prone to collusion attacks and unfair ratings. Therefore to build trust and fairness in the privacy rating system, the evaluation of the indicator should be decentralized, transparent, efficient, secured and automated. Finally, integrating the dynamic rating system within the marketplace framework should introduce minimal overheads to the latency and throughput and not impact the scalability of the marketplace platform.

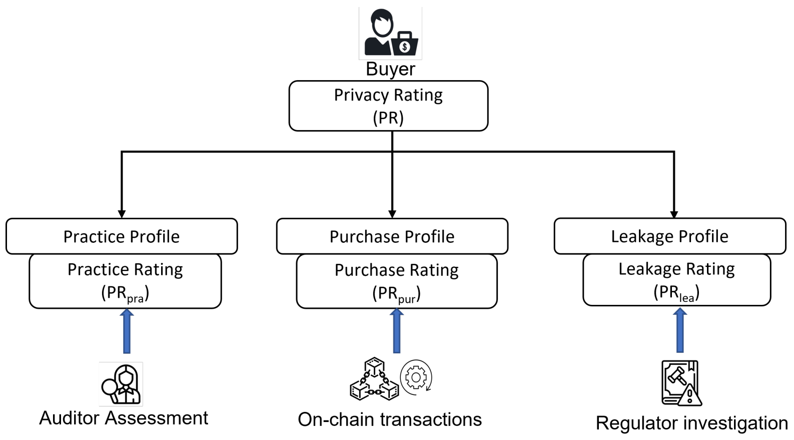


Figure 4.1: Buyer's profiles in the KYBChain to compute their privacy rating

To address the above challenges, we propose KYBChain, a know-your-buyer, blockchain-based marketplace framework integrated with a privacy rating system. Blockchain address the aforementioned issues of manipulation, unfairness and lack of trust by recording buyer's profiles in an immutable, transparent and secure manner. KYBChain leverages smart contracts to relate a buyer's activities to their respective profiles and evaluate their rating in an efficient, decentralized and automated manner. Fig. 4.1 depicts a buyer's practice, purchase and leakage profiles, corresponding ratings and how they are evaluated in KYBChain. Besides users (providers and buyers), auditors and regulators are also the participating actors. An auditor assesses the buyer's practice and security measures against the industry and regulatory standards and validates the buyer's compliance with the disclaimers in their policy. Based on their assessment, they then submits an audit report which becomes the basis of the practice profile of the buyer. The purchase profile tracks all the on-chain provider-specific purchase transactions of the buyer. This work does not consider data purchases outside KYBChain or off-chain trades. The leakage profile is used for data leakage risk assessment of a buyer by monitoring all their purchase transactions and the likelihood of such events in future. When a data leak incident is discovered for a buyer, the regulator investigates the potential damage caused by the incident. they submits an investigation report recorded in the buyer's leakage profile to keep track of their past data leak events. Practice, purchase and leakage profiles are used to evaluate a buyer's practice, purchase and leakage rating.

To the best of our knowledge, we are the first to provide an intuitive and mathematically sound methodology for computing buyers' privacy ratings in the data marketplace. The three characteristics of a buyer include the privacy practice and security measures they has adopted, their data collection pattern and data leak behaviour. These characteristics are rather general, and any model/system that requires data collectors would be able to satisfy them. This chapter makes the following novel contributions:

- We propose KYBChain, a decentralized marketplace framework integrated with a privacy rating system to rate the buyers based on their practice, purchase and leakage profiles. We identify several privacy elements to model these profiles. These identified privacy elements are further used to develop rubrics for scoring the magnitude of risk associated with each profile. Due to the complexity of dependency on these privacy elements and their associated risk on practice, purchase and leakage rating, formulating a single formula is not trivial. Thus, we design different methodologies to convert these risk scores into rating corresponding to each profile. The proposed mechanism offers flexibility and granularity by considering fine-grained details of buyers' characteristics to compute their privacy rating.
- We leverage smart contracts for automation of privacy rating calculation with blockchain transactions and events to ensure the transparency and reliability of the system. By progressively evaluating the privacy rating of a buyer based on their activities in the marketplace over time, KYBChain captures the dynamics of the buyer's behaviour. We incorporate blockchain events as a method to automatically notify participants in case of any privacy violations.

- We conduct multiple simulations to demonstrate the utility of privacy ratings and analyze the effect of rating corresponding to the practice, purchase and leakage profile on the provider's decision-making. We also develop a proof-of-concept implementation of KYBChain in a private Ethereum network and report experimental results regarding gas consumption, throughput and latency. Our results justify the efficacy of privacy rating in aiding providers to make a privacy-aware decision about data sharing. Moreover, our evaluations of KYBChain reveal that the overheads introduced by our mechanism compared to a marketplace that does not incorporate a privacy rating system are insignificant relative to its privacy gains.

The rest of the chapter is structured as follows. The next section briefly discusses the overview of the proposed approach. Section 4.3 explicates the modelling of the privacy rating in more detail. Section 4.4 discusses the proposed decentralized marketplace framework integrated with a privacy rating system. Performance evaluation is presented in section 4.5 and discussion in section 4.6. Finally, section 4.7 concludes the chapter.

4.2 Overview of the Proposed Solution

In this section, we describe the overview of the proposed solution. We present the roles of different participants, the high-level architecture of KYBChain and the privacy rating system.

Table 4.1 lists the notation used in this chapter.

The high-level architecture of KYBChain is illustrated in Fig. 4.2. The participants and their specific roles are explained below:

(i) Providers: Users who possess IoT devices that generate data and wish to monetize their IoT data.

(ii) Buyers: Users who demand and purchase data, e.g. service providers, researchers,

Table 4.1: Table of notation used

Notation	Description
B_i	i^{th} buyer
DP_j	j^{th} provider
$a_k^{ij}(t)$	k^{th} attribute of purchase of B_i where $k \in [1, 6]$
X_k^i	k^{th} privacy element final score
x_k^i	k^{th} privacy element of practice profile of B_i where $k \in [1, 6]$
$A_k^{ij}(t)$	k^{th} privacy element of purchase profile of B_i where $k \in [1, 5]$
$\gamma(t)$	Ageing function for past impact
x_k^{iq}	Assessment of k^th privacy element by q^{th} auditor for B_i
$CP^{ij}(t^-)$	Current data possession of B_i for DP_j data
$R^i(t)$	Data leakage risk assessment for B_i
$\beta(t)$	Decay function for evaluating practice rating
$RepS^i(t)$	Feedback-based reputation score of B_i
$I^i(t)$	Impact of past data leak at time t based on regulator report
CID	IPFS content ID
LF	Leak Factor
$PRV_{lea}^i(t)$	Leakage profile of B_i at time t
$PR_{lea}^i(t)$	Leakage rating of B_i at time t
PR_{lea}^j	Leakage rating preference of DP_j
$Leaked^j(tr^i)$	Leaked data of DP_j corresponding to purchase request of B_i
L^i	Likelihood of B_i
$NormX^i$	Normalized score for practice privacy elements
agr	number of agreements potentially affect or affected
$N_{auditor}$	Number of auditors
N_B	Number of buyers
N_{leak}^i	Number of past data leaks for B_i
pro	Number of providers potentially affect/affected
N^{ij}	Number of purchase agreements between DP_j and B_i
$I_{past}^i(t)$	Overall Impact of B_i past data leaks
$FA^{ij}(t^+)$	Potential future acquisition of B_i to buy DP_j data
I^i	Potential impact of B_i based on their overall data possession
$PRV_{pra}^i(t)$	Practice profile of B_i at time t
$PR_{pra}^i(t)$	Practice rating of B_i at time t
PR_{pra}^j	Practice rating preference of DP_j
$Pref^j$	Preference of DP_j
$PES^{ij}(t)$	Privacy exposure score of DP_j wrt B_i
$PR^{ij}(t)$	Privacy rating of B_i for DP_j
PR^j	Privacy rating preference of DP_j
$PRV^{ij}(t)$	Privacy rating vector of B_i for DP_j
PU_U	Public key of the user. $U \in A, B_i, R$
$Pur^{ij}(t)$	Purchase of B_i at time t to buy DP_j data
$PRV_{pur}^{ij}(t)$	Purchase profile of B_i for DP_j based on their purchases in time period $[0, t]$
$PR_{pur}^{ij}(t)$	Purchase rating of B_i for a time period [0,t]
PR_{pur}^j	Purchase rating preference of DP_j
$Accepted^j(tr^i)$	Purchase request of B_i accepted by DP_j
$tr^{ij}(t)$	Purchase request of B_i to DP_j at time t
$RepS_{DL}^i(t)$	Reputation score of B_i with data leak
S_{imp}	Score for impact attributes
S_{ntmp}	Score for non-temporal privacy elements
S_{pra}	Score for practice privacy elements
S_{tmp}^d	Score for temporal privacy elements of data-type d
sen	Sensitivity of data potentially leak or leaked
Sig_U	Signature of user where $U \in A, B_i, R$
$T_{threshold}^i$	Threshold time
t_{audit}	Time audit assessment was submitted
$t_{purchase}$	Time when buyer purchase provider's data
t_leak	Time when data leak was detected
vol	Volume of data potentially leak or leaked
α_1, α_2, α_3	Weight preference for factors in PES
w_1, w_2, w_3	Weight preference for factors in PR
WTA^j	"Willingness-to-Accept" amount of DP_j
WTP^i	"Willingness-to-Pay" amount of B_i

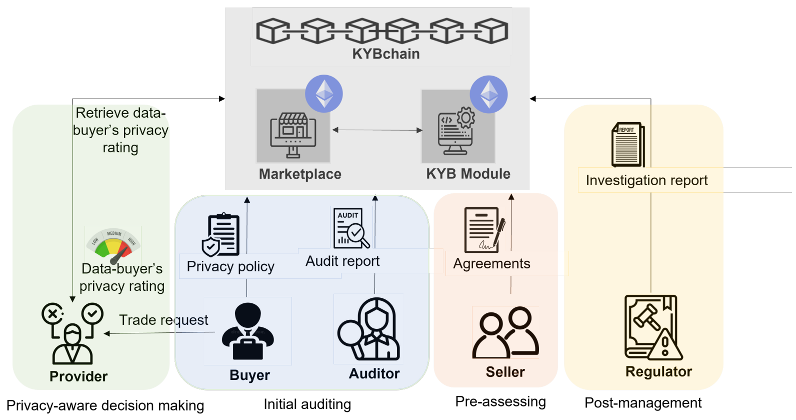


Figure 4.2: High-level architecture of KYBChain

organizations, government, etc. As per many privacy laws [202–204] worldwide, entities that collect personal data must demonstrate how they safeguard this data through their privacy policies. Moreover, to ensure that their practices conform with the stated privacy policies, these entities are obligated to undergo privacy audits [205]. As an industry practice, such entities hire certified public accountants (CPAs) or audit firms to perform privacy audits using auditing procedures such as SOC (System and Organization Controls), developed by the American Institute of CPAs (AICPA) [206]. They receive a SOC certificate showing their compliance with the regulatory requirements. Since the buyer purchases and collects IoT data of providers, we assume the buyer uploads their privacy policy and privacy audit report to the IPFS and updates its address in their user profile in the KYBChain.

(iii) Sellers: Sellers can either be a provider or resellers who play a dual role of buying data from providers and selling purchased data or value-added services to other buyers in the marketplace. We will use seller and provider interchangeably throughout the chapter.

(iv) Auditors: Auditors can be private third-party vendors or government agencies such as FTC. The auditor's responsibilities are to verify the privacy policy and privacy audit uploaded by the buyer, identify vulnerabilities and gaps in the buyer's security measures and submit an assessment report in the KYBChain. We assume auditors registers in the KYBChain and offer a verification service for incentives.

(v) Regulators: Entities responsible for enforcing data protection laws such as OAIC in Australia, DPA in the EU, CCPA in US or ICO in the UK. They oversee data leakage events by providing response guidelines [207] to assist a buyer when leakage occurs. An effective response plan comprises four steps: contain the data leakage, assess the potential harm by gathering evidence, notify the affected individuals and review actions to prevent future occurrence. In KYBChain, all the data leakage incidents are reported to regulators, who investigate the leakage to identify the accountable buyer, determine the cause and assess the potential damage.

KYBChain extends on the MartChain design (see Chapter 3) with the addition of a privacy rating system. KYBChain employs two sets of smart contracts: Marketplace contracts to manage different trading-related functionalities such as user registration, agreement management, and reputation management. KYB-module, devised as a stand-alone application to implement a privacy rating system, leverages several smart contracts to manage buyers' profiles based on their dynamic activities in the marketplace. However, associating all these profiles with the buyer's identity in the blockchain may create a privacy threat for him. This issue can be addressed by unlinking the buyer's identity from the transactions stored in the blockchain, as discussed in [208]. Buyer's profiles are updated during different stages of KYBChain as described below.

- Initial-auditing stage - During this stage, changes in buyer's practices are monitored to assess the buyer's non-compliance risk. This stage is triggered when a newly registered buyer uploads their policy and audit report, or an existing buyer updates their policy and audit report. A notification is sent to the registered auditors, who then validate the buyer's latest policy and audit report. Based on their assessment, auditors submit assessment reports to the KYBChain that updates the buyer's practice profile. The details of the auditor's assessment are explained in section 4.3.1. A successful assessment marks the buyer as KYB-approved and permits them to trade data freely in the marketplace.

- Pre-assessing stage - During this stage, the buyer's provider-specific purchase transactions are monitored to assess the buyer's data accumulation risk. When buyers and providers agree to trade, they deploy an agreement contract in the KYBChain. Based on the buyer's data requirements, the provider adds subscription details in the agreement contract. The agreement contract manages the execution flow of the subscription, starting from initialization, data transfer, receipt and acknowledgement, and payment settlement to finish (see Chapter 3). During the final stage of subscription, i.e., settlement, the buyer's purchase profile is updated based on the subscription specifications as explained in section 4.3.2.

- Post-management stage - During this stage, the buyer's data leak events are monitored and tracked in the buyer's leakage profile. When a data leakage event occurs, a regulator investigates the leakage to identify the accountable buyer and estimate the damage caused by the event. KYBChain can utilize existing data leakage detection mechanisms [209] to identify the buyer responsible for leaking data. A regulator gathers the relevant information about the incident from the responsible buyer, such as system logs [210], to determine the cause and potential harm of the leakage. The regulator then documents their impact analysis and submits the report to the KYBChain that updates the leakage profile of the buyer.

We present the details of smart contracts, transactions and their interactions during different stages of KYBChain in section 4.4. Although KYBChain is built over MartChain, the KYB-module is devised to be agnostic of marketplace implementation. It can be integrated with other blockchain-enabled marketplace solutions, such as distributed product catalogues and access control (see Chapter 2). When a provider receives a new data purchase request from a buyer, the provider can use the buyer's privacy rating vector to make a privacy-aware decision about data sharing. The provider can accept or reject the buyer's request based on their preference. Accordingly, a provider can share their data per the buyer's requested data requirements. To retrieve the buyer's privacy rating, the smart contracts in KYB-module fetch the buyer's practice, purchase and leakage profiles and evaluate the privacy rating as described below:

The privacy rating comprises three components: practice rating, purchase rating and leakage rating corresponding to the buyer's practice, purchase and leakage profiles. It is explicitly evaluated for each provider and for a point in time, which means that (i) for different providers, the buyer may have different privacy ratings, and (ii) the privacy ratings are dynamic. We model the privacy rating of buyer B_i for provider DP_j at time t as a weighted average given in Equation 4.1.

$$PR^{ij}(t) = w_1^j \times PR_{pra}^i(t) + w_2^j \times PR_{pur}^{ij}(t) + w_3^j \times PR_{lea}^i(t) \tag{4.1}$$

where w_1^j, w_2^j, and w_3^j are the weighting factors for the ratings corresponding to their buyer's different profiles. The choice of these weighting factors depends on the provider's preferences. $PR_{pra}^i(t)$, $PR_{pur}^{ij}(t)$ and $PR_{lea}^i(t)$ are the practice, purchase, and leakage ratings of the buyer. Each rating $PR_{pra}^i(t)$, $PR_{pur}^{ij}(t)$ and $PR_{lea}^i(t)$ are in the range from 0 to 1; hence, $PR^{ij}(t) \in [0,1]$. Note that the purchase profile captures the provider-buyer interactions. Hence, PR_{pur}^{ij} is a provider-dependent rating computed for each pair. The basic premises for the calculation of these components are: (i) better privacy practices and security measures reduce the provider's privacy risks and increase their $PR_{pra}^i(t)$; (ii) higher the sensitivity of accumulated data greater the provider's privacy risks and lower their $PR_{pur}^{ij}(t)$; and (iii) higher the probability and harm of data leakage event, greater the provider's privacy risks and lower their $PR_{lea}^i(t)$. As presented in [211], each provider differs in their extent of privacy expectations and attitude, as some are privacy fundamentalists while others are privacy pragmatists or unconcerned. Therefore, to enable fine-grained characterisation of providers' privacy preferences, we have structured the privacy rating vector $PRV^{ij}(t)$ of a buyer using the following tuple:

$$PRV^{ij}(t) =< PR^{ij}(t), PR_{pra}^i(t), PR_{pur}^{ij}(t), PR_{lea}^i(t) > \tag{4.2}$$

We present buyer's practice, purchase and leakage profiles as $PRV_{pra}, PRV_{pur}, PRV_{lea}$ and they are represented as a tuple of the privacy elements presented in the Sections 4.3.1, 4.3.2 and 4.3.3 respectively. Our approach consolidates these privacy elements, as depicted in Fig. 4.3, that form the core of the modelling of the profile and formulation of the corresponding rating, as explained next.

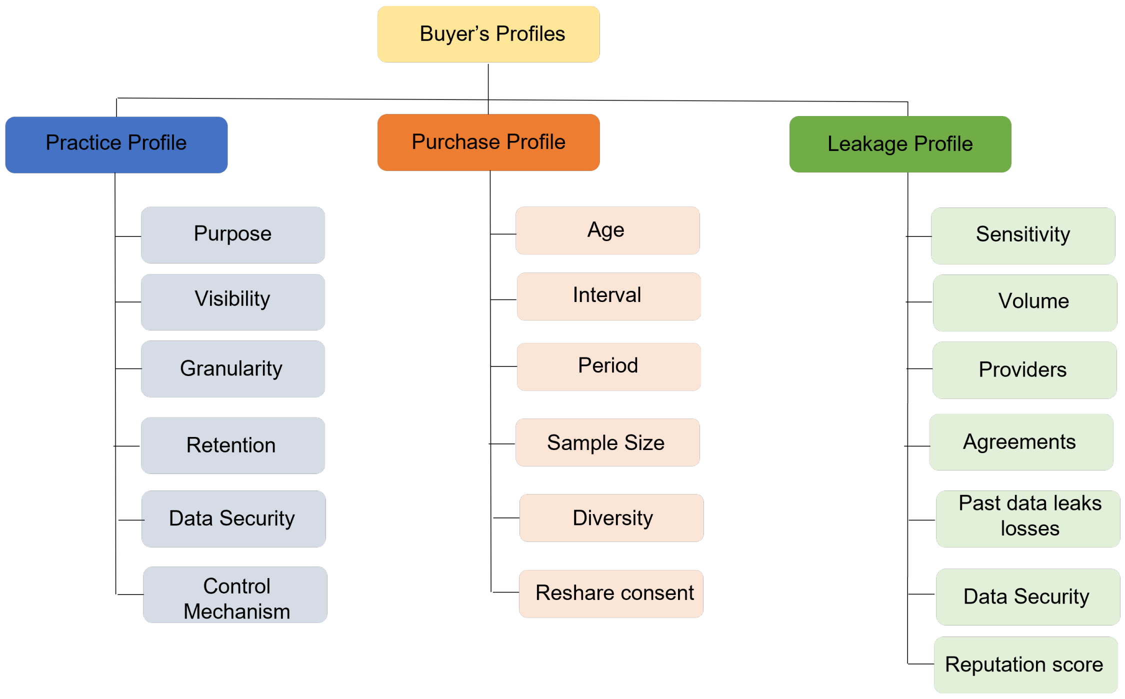


Figure 4.3: Privacy elements to model buyer's practice, practice and leakage profiles

4.3 Modelling of Privacy Rating components

In this section, we describe the modelling of each profile and the formulation of the corresponding rating.

4.3.1 Practice profile

Recall from section 4.2 when a data leak occurs, a regulator investigates the data leak to determine the impact. These assessment reports are visible to all the participants in the KYBChain, hence, improving the transparency of compliance and data protection practices adopted by the buyers. We assume privacy-unconcerned providers will be motivated only by economic benefits. In contrast, privacy-concerned providers may prefer to sell their data to those buyers who have adopted good practices and measures for safeguarding their privacy. Therefore, we propose a rating system that privacy-concerned providers could utilize such that the better the privacy practices and security measures adopted by the buyer, the higher their practice rating. The practice rating is buyer-specific and evaluated for a point in time, as explained later.

To model the buyer's practice profile, we first identify the privacy elements to determine the buyer's practice and security measures. Next, we develop a rubric that auditors will use to score these privacy elements against the industry standards and regulatory requirements. Finally, these scores are used to evaluate the buyer's practice rating.

Table 4.2 presents the identified privacy elements [212] that form the basis of the practice profile. The privacy elements are not limited to the list in the table; a new privacy element can be added to model the practice profile.

Privacy Elements	**Description**
Purpose	This factor determines how the purchased data will be used. • Single: Purchased data can only be used once for a one-time single purpose. • Reuse: Purchase data could be reused for the same purpose more than once, for a selected set of explicitly defined purposes, or unforeseeable related purposes. • Any: Purchased data could be used in any way.
Visibility	This factor determines who is permitted to access or use the purchased data. • Authorized party: Only the buyer who has bought the data is permitted to access/view the purchased data. • Third party: This determines if the purchased data is shared with third parties for legitimate purposes. Disclosing purchased data to third parties is risky since they are not abiding by any agreement with the providers. However, the risk can be controlled if the privacy policy provides the details about the third party, such as name and location (country), details of the shared data, the purpose of sharing, duration of retention of shared data, security measures, etc. • All: This dimension represents the least amount of privacy protection since the purchased data can be offered to anyone.
Retention	This factor determines how long the buyer can retain the purchased data. Data that has passed its expiry time must be deleted, or a renewed agreement must refresh the retention period. Data retention impacts providers' privacy preferences based on whether the data can be retained for a short duration, long duration or stored at the buyer's end for an indefinite duration.

Granularity	This factor determine the level of data precision required by the buyer. • Existential: This dimension corresponds to the existence of the data instead of the actual data value. • Partial Precision: This dimension corresponds to the data value altered in a non-destructive way using techniques like aggregation or summarization, differential privacy etc., to reduce preciseness to preserve the provider's privacy • Full precision: This dimension corresponds to the actual data without alteration. Precise data could uniquely identify an individual and vice versa.
Data security	This factor determines the buyer's ability to protect the provider's data from misuse, interference and loss and unauthorised access, modification or disclosure. This factor is determined based on the communication, storage and code security mechanism followed by the buyer. • Communication Security: This determines if all communication channels that transmit IoT data are secure using SSL/TLS. Communication security can range from insecure channels to weak communication protocols to strong communication security using mutual authentication and encryption. • Code Security: This determines if the buyer's application or software demonstrates secure coding practices such as audit logs are maintained, privileges are carefully planned, hash functions and cryptography are carefully designed etc. It ranges from no code security to few secure coding practices adopted to good secure coding practices. • Storage Security: This determines if the data storage has adequate protection mechanisms, including encryption, authentication, authorisation and security software such as anti-viruses, intrusions detection systems etc. Storage security can range from no measures to weak to strong security measures that guarantee confidentiality, integrity and availability.

Control Mechanism	This factor determines if providers can perform the following actions on their sold data: view, update, or delete (i.e., if the provider has the 'right to be forgotten). Depending on the level of control given to the provider over the data, the dimensions of the control mechanism can range from no control to partial control to complete control over viewing, updating and deletion.

Table 4.2: Identified privacy elements to model the buyer's practice profile

These privacy elements are further used to develop a rubric for scoring the buyer's practice and security measures. To keep the system simple and understandable for users, we adopt a 5-point Likert scale [213] in $[-1, 1]$ to score each privacy element based on the following criteria:

1. When the buyer's practice and security measure corresponding to the privacy element is worst, a score of -1 is given.
2. When the buyer's practice and security measure corresponding to the privacy element is bad, a score of -0.5 is given.
3. When the buyer's practice and security measure corresponding to the privacy element is average, then a score of 0 is given.
4. When the buyer's practice and security measure corresponding to the privacy element is good, a score of 0.5 is given.
5. When the buyer's practice and security measure corresponding to the privacy element is best, then a score of 1 is given.

An illustrative example of such a rubric is given in Fig. 4.4. It can be designed as per industry standards or regulatory requirements. Next, we model the practice profile ($PRV^i_{pra}(t)$) of B_i at time t as a tuple given by

$$PRV^i_{pra}(t) =< S_{pra}(x^i_1), S_{pra}(x^i_2), S_{pra}(x^i_3), S_{pra}(x^i_4), S_{pra}(x^i_5), S_{pra}(x^i_6) > \tag{4.3}$$

where x_1^i, x_2^i, x_3^i, x_4^i, x_5^i, x_6^i are the privacy elements, i.e., purpose, visibility, retention, granularity, data security and control mechanism, respectively. $S_{pra}()$ is the score as per the rubric guide.

		$S_{pra}(x_k)$				
		-1	-0.5	0	0.5	1
Privacy Elements	Purpose (x_1)	Reuse for any purpose	Reuse for a selected set of related purposes	Not applicable	Reuse for a selected set of purposes	Single purpose only
	Visibility (x_2)	Sharing with anyone	Sharing with third-party without details	Not applicable	Third-party with details provided	Authorised Party
	Retention (x_3)	Data is retained indefinitely	Data is retained for a long period of time	Not applicable	Data is retained for a short period of time	Single use
	Granularity (x_4)	Specific data with high precision	Partial precision data	Not applicable	Low precision data	Existential data
	Data Security (x_5)	No security measures are applied	Security measures exist but are weak and unsatisfactory	Not applicable	Security measures can be improved	High security measures guaranteed
	Control Mechanism (x_6)	None	Data can be viewed	Not applicable	Data can viewed and partial control over update and delete	Complete control over view, update and delete

Figure 4.4: Rubric for scoring privacy elements of buyer's practice and security measures

As discussed in section 4.2, the auditor assigns these scores based on their assessment of the buyer's privacy policy and audit report. However, a malicious buyer can collude with the auditor to manipulate the assessment in their favour. This creates a false sense of security and may lead to promulgating policies inconsistent with actual practices [214] of the dishonest buyer. Therefore, instead of relying on a single auditor, we assume multiple auditors perform the assessment who are randomly selected to prevent a collusion attack. The final score of each privacy element is determined by averaging the score in the assessment reports submitted by all the auditors. Suppose $N_{auditor}$ auditors submitted their assessment report for a buyer B_i. The score for q^{th} privacy element given by q^{th} auditor is $S_{pra}(x_k^{iq})$ where $q \in [1, N_{auditor}]$ and $k \in 1, 6$. Equation 4.4 gives the final score for each privacy element.

$$X_k^i = \frac{1}{N_{auditor}} \sum_{q=1}^{N_{auditor}} S_{pra}(x_k^{iq}) \tag{4.4}$$

Next, we compute the normalized score ($NormX^i$) as the average of all the privacy element's final scores given by:

$$NormX^i = \frac{1}{6} \sum_{k=1}^{6} X_k^i \tag{4.5}$$

where, X_k^i is the final score for k^{th} element. $NormX^i$ is a real number in the range of $[-1, 1]$. The buyer who follows the best privacy practice receives the maximum normalized score, i.e., $NormX = 1$, while the buyer who adopts poor practices receives the minimum normalized score, i.e., $NormX = -1$.

We adopt the rating concept in [100] to calculate the practice rating, in which the logistic function is used to map the risk scores to the rating. As in real-life scenarios, a buyer's practice and security measures will be outdated with time due to the rapidly changing threat landscape or newly discovered vulnerabilities in cybersecurity. Hence, to ensure the buyer's practice and measure become obsolete with time, we introduce a time threshold. This means that the buyer's privacy assessment remains effective for a specific time threshold, and so does their practice rating. Beyond this time, their practice rating will decay, reflecting outdated practices and measures. The decline in practice rating will compel the buyer to periodically review, update and audit their privacy practice and security measures as per the recommended industry practice [214].

The normalized score ($NormX^i$) of B_i computed in Equation 4.5 is used to map to their practice rating $PR^i_{pra}(t)$ as given by Equation 4.6.

$$PR^i_{pra}(t) = \begin{cases} \frac{1}{1+e^{-5NormX^i}} & t \leq T^i_{threshold} \\ \frac{1}{1+e^{-5NormX^i}}\beta(T^i_{threshold} - t) & t > T^i_{threshold} \end{cases} \tag{4.6}$$

where $T^i_{threshold}$ is the threshold time that enables practice rating to decay in time. It is determined based on the time and periodicity of the audit. $\beta(T^i_{threshold} - t)$ is the decay factor that can be realized using an exponential function such as $\beta(t) = e^{-f(t)}$.

4.3.2 Purchase profile

In the marketplace, a provider can accept different demands from the buyer and disclose their data per the buyer's requirements. Recall from section 4.1 a buyer accumulating all these disclosed data may cause predictive harm that can aggravate the provider's privacy risk over time. Hence, inadvertently exposing the provider's privacy [215]. The purchase profile of a buyer is calculated based on the provider-specific data purchases to determine the provider's privacy exposure. We assume that privacy-concerned providers may prefer to sell their data to those buyers for whom providers' privacy exposure is low. Therefore, we propose a rating system that privacy-concerned providers could utilize, such that the lower their privacy exposure to buyers, the higher their purchase rating. The purchase rating is provider-buyer specific and evaluated for a period of time, as explained later.

We first identify the privacy elements to model the buyer's purchase profile. Next, we develop a rubric to score these privacy elements based on the provider's severity of privacy risk associated with all the provider's data the buyer has purchased. Finally, we describe the steps to evaluate the provider's privacy exposure from these scores and map it to the buyer's purchase rating.

As suggested in [102], the privacy risk due to prolonged data sharing depends on the sensitivity of the disclosed data. Sensitive data reveal confidential and private information about the provider. The sensitivity of IoT data [176] is the function of temporal and non-temporal factors. The temporal factors are based on the data specifications preferred by the buyer for their usage: data age, data sampling interval, and data period. The non-temporal factors depend on the overall purchase behaviour of the buyer regarding the provider: purchase diversity and purchase sample size. Table 4.3 presents the description of the identified privacy elements to model the purchase profile.

Privacy Elements	**Description**
Data age	This factor is defined as the amount of time elapsed between when the data got generated and when the buyer collected it. Data may be collected shortly after generation, i.e., near real-time data. Data may be generated in the past and purchased later, i.e., archived data. Availability, accuracy and relevancy decrease with data age. The older the data, the less likely it is to cause harm, reducing the privacy risk for the provider.
Data Sampling interval	The sampling interval refers to the time between the measurements of two data samples. The shorter the data sampling interval, the higher the amount of sensitive information it reveals, creating new privacy threats to the provider [216].
Data period	Period refers to the interval length within which all the data samples are measured. Data analysis, measured and collected over an extended period, may reveal different trends over time and expose sensitive characteristics related to the provider's health, habits, interests, and others, causing increased privacy threats.

Diversity	This element refers to the variety of the data types that a buyer has purchased from different IoT devices of a provider. High diversity correlates with privacy risk since a more accurate provider profile can be generated by linking their diverse data to their habits, movements, and emotions.
Sample size	Sample size refers to the total number of provider's data samples a buyer has purchased. Suppose a buyer has bought GPS data from a provider multiple times. The buyer, acting in bad faith, might link all the GPS datasets over time and, thus, increases the sample size. The combined dataset encompasses increasingly long periods, enabling insights that can only be derived from large datasets. Sample size usually depends on the period and interval but to consider the scenario in which data samples may be lost during transmission, we have a separate parameter.

Table 4.3: Identified privacy elements to model purchase profile of a buyer

To model a purchase profile using these privacy elements, we first define a purchase. In the marketplace, a purchase corresponds to disclosing the provider's data to the buyer as per their requirement. A buyer's data requirement comprises the following attributes: data type, age, period, and interval. For each purchase, a provider can explicitly consent the buyer to share their data with other buyers for monetary benefits. Provider's data for which reshare consent is *true* may spread to different buyers in the marketplace. We represent a purchase ($Pur^{ij}(t)$) that the buyer (B^i) made at time t to buy the provider's (DPj) data as a tuple given by Equation 4.7.

$$Pur^{ij}(t) =< a_1^{ij}(t), a_2^{ij}(t), a_3^{ij}(t), a_4^{ij}(t), a_5^{ij}(t), a_6^{ij}(t) > \tag{4.7}$$

where $a_1^{ij}(t)$, $a_2^{ij}(t)$, $a_3^{ij}(t)$ represent temporal attributes, age, interval and period, respectively. $a_4^{ij}(t)$ and $a_5^{ij}(t)$ represent non-temporal attributes, data type and sample size, respectively. $a_6^{ij}(t)$ represent the reshare consent as either *true* or *false*. These attributes are specified in the subscription agreed upon between buyer and provider as discussed in section 4.4.1.

A provider is generally aware of all the data the buyer has bought directly from him. Besides, the buyer can also accumulate data from other sources without the provider's

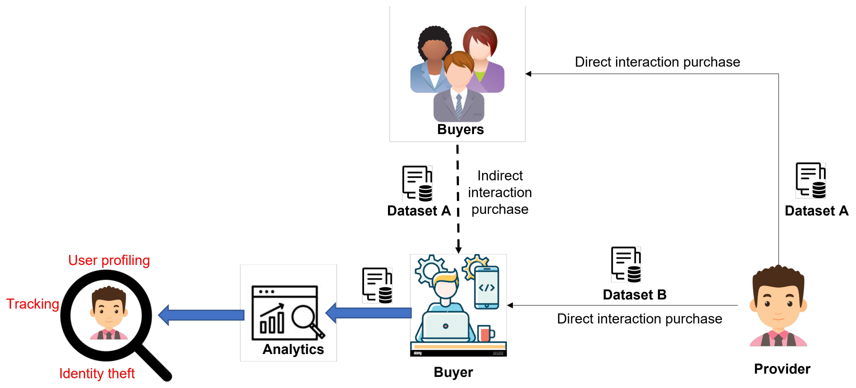


Figure 4.5: Direct and indirect interaction to purchase the provider's data

knowledge. These sources can be other buyers who purchased the provider's data and were given permission by the provider to reshare their data. Therefore, we consider two ways of interactions as depicted in Fig. 4.5 through which buyers can purchase the provider's data. Direct interaction means the buyer purchases the provider's data directly from him. Indirect interaction means the buyer purchases the provider's data from other buyers.

The purchase profile is calculated based on all the purchases via direct or indirect interactions over a period of time. We model the purchase profile of B^i for DP^j as a tuple given by Equation 4.8:

$$PRV^{ij}_{pur}(t_n) =< A^{ij}_1(t_n), A^{ij}_2(t_n), A^{ij}_3(t_n), A^{ij}_4(t_n), A^{ij}_5(t_n) > \tag{4.8}$$

where $A^{ij}_1(t_n)$, $A^{ij}_2(t_n)$, $A^{ij}_3(t_n)$ corresponds to temporal privacy elements, i.e., age, interval and period respectively. $A^{ij}_4(t_n)$ and $A^{ij}_5(t_n)$ correspond to non-temporal privacy elements, i.e., diversity and sample size, respectively. We derive privacy elements of $PRV^{ij}_{pur}(t_n)$ based on all the purchases ($Pur^{ij}(t)$) of B^i to buy DP_j data in the time period $t \in [t^i_0, t_n]$ as explained next.

These privacy elements are further used to develop a rubric for assigning a risk score to the buyer's purchase attributes. We adopt a 5-point scale in $[-1, 1]$ similar to the practice profile to score each attribute based on the following criteria:

1. When the risk associated with the attribute is very high, a score of -1 is given.

2. When the risk associated with the attribute is high, a score of -0.5 is given.
3. When the risk associated with the attribute is average, a score of 0 is given.
4. When the risk associated with the attribute is low, a score of 0.5 is given.
5. When the risk associated with the attribute is very low, a score of 1 is given.

We develop different methodologies to evaluate temporal and non-temporal privacy elements in purchase profile. This is because, in the case of temporal, the privacy risk is associated with each temporal attribute of purchase. For example- as the sampling interval decreases, the identifiability of the data increases. GPS data collected daily will likely reveal the providers' places of residence or employment. Data collection on an hourly basis will likely reveal individuals' residences, places of employment, children's school locations, and relatives' homes. Sampling interval decreasing to minute-by-minute samples will enable an analysis of behaviour, such as inferences about providers' walking and visiting patterns or shopping habits. Therefore, to determine the temporal privacy elements, we first assign a risk score to the temporal attribute of each purchase and then aggregate them for all the purchases. While in the case of non-temporal, risk is associated with the data accumulation for overall purchases. For instance- an individual piece of data may seem innocuous, e.g., step counts are not considered sensitive. However, with the increase in the volume of data collected from different diverse IoT devices, buyers can learn powerful patterns of providers' preferences and behaviours, e.g., the data from smartwatches and fitness trackers can infer providers' physical activities such as walking, running, and jumping, with high accuracy. Therefore, to determine the non-temporal privacy elements, we first aggregate non-temporal attributes for all the purchases and then assign risk scores, as explained next.

Temporal Elements: As suggested in [163], the risk posed by different types (e.g. GPS and temperature) of IoT data varies. The risk magnitude depends on how much sensitive information a particular data type reveals about the provider [217]. Therefore, a separate rubric score guide for each data type should be designed to map temporal attributes of different data types on the same scale of risk magnitude. Fig. 4.6 illustrates an example of

		$S^{GPS}_{tmp}(a_k)$				
		-1	**-0.5**	**0**	**0.5**	**1**
Temporal Elements	**Data Age (a_1)**	Real time	0-24hrs	1 – 30 days	1 – 12 month	>1 year
	Sampling interval (a_2)	<60sec	1-60min	1 – 12hrs	12-24hrs	>24hrs
	Data Period (a_3)	> 7 days	1 – 7 days	12-24 hrs	1-12hrs	<1hr

Figure 4.6: An illustrative example of rubric to determine temporal privacy elements of GPS data.

a rubric score guide for the temporal specification of GPS data. Since temporal elements are a function of time, the associated harm to the provider's privacy decreases with time. Therefore, an ageing factor (e.g. $exp^{-(t_n-t)}$) is used to give more relevance (higher weights) to the attributes of a recent purchase and less relevance (lower weights) to past purchase attributes. We formulates the k^{th} temporal privacy element ($A^{ij}_k(t_n)$ where k=1,2,3) of the purchase profile of B_i given by Equation 4.9.

$$A^{ij}_k(t_n) = \frac{1}{N^{ij}(t_n)} \sum_{t=t^i_0}^{t_n} S^d_{tmp}(a^{ij}_k(t))exp^{-(t_n-t)} \tag{4.9}$$

where $N^{ij}(t_n)$ is the total number of purchase transactions by B_i with DP_j in the time period $[t^i_0, t_n]$. S^d_{tmp} is the temporal rubric score guide for data-type d.

Non-temporal Elements: Algorithm 2 gives the steps to compute the non-temporal element in the purchase profile. First, we count the unique values in each non-temporal attribute of all the purchases. We then assign a risk score to these counts based on the aforementioned risk criteria to determine the non-temporal elements of the purchase profile. An illustrative example of a rubric to score the non-temporal attributes is depicted in Fig. 4.7.

Next, we describe the steps to evaluate the provider's privacy exposure based on all their disclosed data in the marketplace.

Provider's privacy exposure depends on the risk posed by their overall disclosed data in the marketplace. Therefore, when a provider receives a new data purchase request from a buyer, we determine the provider's privacy exposure score (PES) based on the risk posed

Algorithm 2 Algorithm to determine non-temporal privacy elements in purchase profile

```
Input: $Pur^{ij}(t)\ \forall\ t \in [t_0^i, t_n]$, $S_{ntmp}$
Output: $A_4^{ij}(t_n)$, $A_5^{ij}(t_n)$
1: initialize empty list $diversity$, $samples = 0$
2: for each $Pur^{ij}(t)$ in time $[t_0^i, t_n]$
3:     if $a_4^{ij}(t) \notin diversity$
4:         append $a_4^{ij}(t)$ to $diversity$
5:     samples $+=$ $a_5^{ij}(t)$
6: $A_4^{ij}(t_n) = S_{ntmp}(\text{length}(diversity))$
7: $A_5^{ij}(t_n) = S_{ntmp}(samples)$
8: return $(A_4^{ij}(t_n), A_5^{ij}(t_n))$
```

		$S_{ntmp}(a_k)$				
		-1	-0.5	0	0.5	1
Non Temporal Elements	Diversity (a_4)	>20	15-20	10-15	5-10	<5
	Sample size (a_5)	>10000	5000-10000	1000-5000	500-1000	<500

Figure 4.7: Rubric to determine the non-temporal privacy element of a purchase. This guide is generated considering all the purchases of the buyer specific to a provider.

by (i) the data requested in the latest purchase request, (ii) the data currently possessed by the buyer, (iii) the buyer can potentially acquire from other buyers. We use an example scenario as depicted in Fig. 4.8 to explain the aforementioned factors to determine the provider's PES. Suppose there are 2 buyers (B_1 and B_2) and a provider (DP). B_1 purchased DP's data (d_1^1 and d_2^1). B_2 purchased DP's data (d_1^2 and d_2^2). DP has only given reshare consent to d_1^2. Now assume that B_1 sends a new purchase request to buy d_3^1 data. DP's privacy exposure score depends on the risk posed by B_1's current possession, i.e., d_1^1 and d_2^1, their latest purchase request, i.e., d_3^1, and potential future acquisitions, i.e., d_1^2. Next, we formulate the provider's PES and map it to the buyer's purchase rating based on the following premises: the higher the provider's PES, the higher the privacy risk of disclosing data to the buyer the lower the buyer's purchase rating.

Suppose there are N_B buyers in the marketplace. We assume that each buyer B_m where $m \in [1, N_B]$ purchased provider's (DP_j) data in the time period $t = [t_0^m, t_{n-1}]$. At the time t_n, B_i sends a new purchase request to DP_j specifying their data requirement as (d, $a_1^{ij}(t_n)$, $a_2^{ij}(t_n)$, $a_3^{ij}(t_n)$) where d is the data type. $a_1^{ij}(t_n)$, $a_2^{ij}(t_n)$, $a_3^{ij}(t_n)$ are the temporal

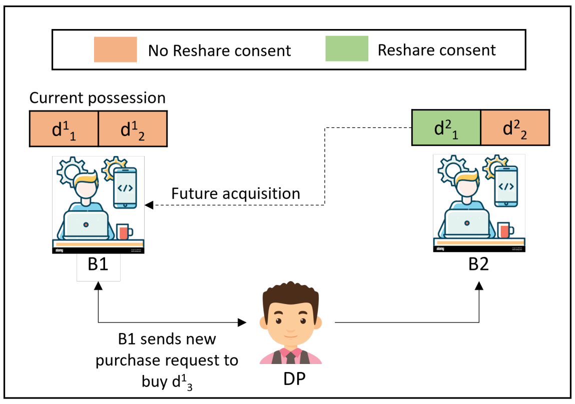


Figure 4.8: An example scenario to illustrate the factors used in evaluating the provider's privacy exposure score

attributes data age, data sampling interval, period. $a^{ij}_4(t_n) = d$ and $a^{ij}_5(t_n) = \frac{a^{ij}_3(t_n)}{a^{ij}_2(t_n)}$ are the bon-temporal attributes diversity and sample size.

The risk posed by the latest data purchase request is represented as a tuple given by Equation 4.10.

$$tr^{ij}(t_n) =< A^{ij}_1(t_n), A^{ij}_2(t_n), A^{ij}_3(t_n), A^{ij}_4(t_n), A^{ij}_5(t_n) > \tag{4.10}$$

where $A^{ij}_k(t_n) = S^d_{tmp}(a^{ij}_k(t_n))$ for $k = 1, 2, 3$ and $A^{ij}_k(t_n) = S_{ntmp}(a^{ij}_k(t_n))$ for $k = 4, 5$.

The risk posed by current data possession is represented as a tuple given by Equation 4.11.

$$CP^{ij}(t^-_n) =< A^{ij}_1(t^-_n), A^{ij}_2(t^-_n), A^{ij}_3(t^-_n), A^{ij}_4(t^-_n), A^{ij}_5(t^-_n) > \tag{4.11}$$

where $A^{ij}_k(t^-_n)$ is the k^{th} element in the buyer's purchase profile based on all the purchases made between time $[t^i_0, t_{n-1}]$.

The risk posed by the potential data they can acquire from other buyers is represented as a tuple given by Equation 4.10.

$$FA^{ij}(t^+_n) =< A^{ij}_1(t^+_n), A^{ij}_2(t^+_n), A^{ij}_3(t^+_n), A^{ij}_4(t^+_n), A^{ij}_5(t^+_n)) > \tag{4.12}$$

where

$$A^{ij}_k(t^+_n) = \frac{1}{(N_B - 1)} \sum_{\substack{m=1 \\ m \neq i}}^{N_B} \overline{A^{mj}_k}(t_{n-1}) \tag{4.13}$$

where $\overline{A_k^{mj}}(t_{n-1})$ is the privacy elements based on the purchases of B_m between the time $[t_0^m, t_{n-1}]$. These are computed using the same methodology for temporal and non-temporal elements as discussed above; however, only purchases for which reshare consent, i.e., $a_6^{mj}(t) = \text{true}$, are considered.

Depending on different implementations or applications, the privacy elements of these factors can be combined to represent the complex relationships between them. Risk assessment based on these factors can be performed using rules [218] that are generally defined based on relevant knowledge and opinion from human experts or much more complex fuzzy logic models [219, 220]. However, we use a simplified approach of weighted average to evaluate the provider's privacy exposure score as given by Equation 4.14:

$$PES^{ij}(t_n) = \frac{1}{5}\sum_{k=1}^{5} \alpha_1 \times A_k^{ij}(t_n) + \alpha_2 \times A_k^{ij}(t_n^-) + \alpha_3 \times A_k^{ij}(t_n^+) \tag{4.14}$$

where $PES^{ij}(t_n)$ is a real number in $[-1, 1]$. α_1, α_2 and α_3 are the weights based on the provider's preference such that (α_1, α_2 and $\alpha_3 \geq 0$) and (α_1+ α_2+ $\alpha_3 = 1$).

The privacy exposure score $PES^{ij}(t_n)$ computed in Equation 4.14 for DP_j with respect to B^i is then mapped to the buyer's purchase rating $PR_{pur}^{ij}(t_n)$ using the logistic function as given by Equation 4.15.

$$PR_{pur}^{ij}(t_n) = \frac{1}{1 + e^{-5PES^{ij}(t_n)}} \tag{4.15}$$

where $PR_{pur}^{ij}(t_n)$ is a real number in the range $[0, 1]$.

4.3.3 Leakage Profile

A leakage profile is used to assess the buyer's data leak risk by quantifying their potential damage and likelihood of a data leak event. It helps providers minimize data leak risk by deciding and controlling data sharing as necessary. To this end, we propose a rating system that depends on the data leak risk assessment satisfying the following premises: the higher the data leak risk, hence, the lower the reliability of the buyer's capability in the provider's eye, the lower their leakage rating. We compute the leakage rating specific to a buyer at a particular time.

To model the leakage profile, we first identify the privacy elements for the leakage profile. We then quantify these elements to assess the data leak risk and map this risk assessment to the buyer's leakage rating.

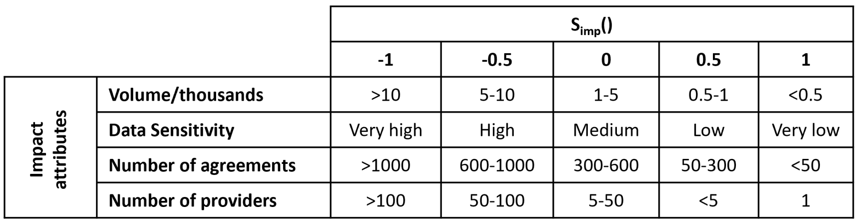
Risk assessment [221] involves the calculation of (i) the impact, which is the level of damage that will be incurred, and (ii) likelihood which is the probability that an event will occur. To make the data leak risk assessment more systematic, we decompose the impact and likelihood into more fine-grained elements as explained below:

$$\text{Risk} = \text{Impact} \times \text{Likelihood} \tag{4.16}$$

Impact Quantification: An impact is quantified based on the damage that a data leak event might cause to the provider, along with the financial loss and reputation damage that a buyer might suffer [81]. We quantify the impact by using four different attributes: (i) volume (*vol*) and (ii) sensitivity (*sen*) of data that buyer can potentially leak, which indicates the extent of possible harm to providers; (iii) the number of agreement (*agr*) violation that reflects the magnitude of potential financial loss for a buyer; (iv) number of providers (*pro*) that might get affected causing potential reputation damage to the buyer. We designed a similar scoring system as suggested in [222] to determine the combined influence of all these attributes on the potential impact. An illustrative example of the scoring system is given in Fig. 4.9. Suppose a data leak incident is reported against a buyer B_i with estimated impact given as 10,000 datasets of highly sensitive data affecting 40 providers and violating 100 agreements. In this case, $S_{imp}(vol)$, $S_{imp}(sen)$, $S_{imp}(agr)$ and $S_{imp}(pro)$ as per the illustrative example given in Fig. 4.9 is -1, -0.5, 0.5, 0 respectively. We adopt the method in [222] to evaluate the combined impact (I) of all these attributes, which is formally given by:

$$I = \frac{v_1 \times S_{imp}(vol) + v_2 \times S_{imp}(sen) + v_3 \times S_{imp}(agr) + v_4 \times S_{imp}(pro)}{v_1 + v_2 + v_3 + v_4} \tag{4.17}$$

where $S_{imp}(vol)$, $S_{imp}(sen)$, $S_{imp}(agr)$ and $S_{imp}(pro)$ represents the score of volume, sensitivity, agreements and providers as per the Fig. 4.9 respectively. v_1, v_2, v_3, and v_4



		$S_{imp}()$				
		-1	-0.5	0	0.5	1
Impact attributes	Volume/thousands	>10	5-10	1-5	0.5-1	<0.5
	Data Sensitivity	Very high	High	Medium	Low	Very low
	Number of agreements	>1000	600-1000	300-600	50-300	<50
	Number of providers	>100	50-100	5-50	<5	1

Figure 4.9: An illustrative example of a scoring system to score the severity of the impact attributes

represent the corresponding provider's weightage preference for each parameter. I is in the range of $[-1, 1]$. This means that when I is -1, the potential impact of a data leak is low, and when I is 1, the potential impact is high.

Likelihood Quantification: We define the likelihood as how likely a buyer leaks the provider's purchased data to a third party. A data leak can be caused intentionally or inadvertently [223]. Intentional data leak is caused deliberately because of the malicious behaviour of the buyer. A low-reputed buyer tends to intentionally share the data with unauthorized parties [223] for monetary purposes. Therefore, we assume that the likelihood of an intentional data leak depends on the buyer's reputation score. As described in [224], risk analysis usually involves the quantification of probabilities of a loss event based on experience and the costs of the associated loss. Therefore, KYBChain employs a reputation mechanism that considers interaction feedback (similar to the one discussed in Chapter 3) and data leak events to calculate the buyer's reputation score. To incorporate the past data event in evaluating the reputation score, we define a leak factor (LF) that solely depends on the impact of all buyer's past data leak events. We formulate the new reputation score of B_i at time t_n as given by Equation 4.18. The value of LF lies in $[0, 1]$ and is given by Equation 4.19. With the increase in the overall impact of the past leak events, LF decreases, thus decreasing the buyer's reputation score.

$$RepS^i_{DL}(t_n) = RepS^i(t_n).LF(t_n) \tag{4.18}$$

where

$$LF(t_n) = 2 - 2^{I^i_{past}(t_n)} \tag{4.19}$$

where $RepS^i(t_n)$ is the buyer's feedback-based reputation score, $RepS^i_{DL}(t_n)$ is the buyer's reputation score incorporating their past data leak events, $I^i_{past}(t_n)$ is the overall impact of all the buyer's data leak events happened in the past and is calculated as explained next.

We formulate Equation 4.20 to determine the overall impact ($I^i_{past}(t_n)$) of all the past data leak events of the buyer. We consider the damage due to recent data leak events more relevant than the previous data leak events. Therefore, we introduce an ageing function ($\gamma^{(t_n-t)}$ such that $0 < \gamma < 1$), which assigns higher weights to the impact of recent data leak events and lower weights for past events.

$$I^i_{past} = \frac{\sum_{t=t_1}^{t_l} I^i(t)\gamma^{(t_n-t)}}{N^i_{leak}} \tag{4.20}$$

where $I^i(t)$ is the impact of the data leak event that occurred at time t such that $t < t_n$. N_{leak} is the number of past data leak events for the buyer. $I^i(t)$ is calculated using Equation 4.17 based on the regulator investigation of the data leak event as explained in section 4.4.

Furthermore, the failure of buyers to implement adequate security systems [189] can cause an accidental data leak. Since the accidental data leak results from poor security practices followed by the buyer, we assume the likelihood of an inadvertent data leak depends on the buyer's data security score, as discussed in section 4.3.1. We assign the buyer's likelihood that a data leak event will occur based on their $RepS^i_{DL}$ and data security score ($S_{pra}(x^i_5)$) as given by Equation 4.21.

$$L^i = \begin{cases} 0.80 & RepS^i_{DL} \leq 0.5 \ \& \ S_{pra}(x^i_5) \leq 0 \\ 0.60 & RepS^i_{DL} \leq 0.5 \ \& \ S_{pra}(x^i_5) > 0 \\ 0.40 & RepS^i_{DL} > 0.5 \ \& \ S_{pra}(x^i_5) \leq 0 \\ 0.20 & RepS^i_{DL} > 0.5 \ \& \ S_{pra}(x^i_5) > 0 \end{cases} \tag{4.21}$$

where $RepS^i_{DL} \in [0, 1]$ and $S_{pra}(x^i_5) \in [-1, 1]$. We assume that the maximum likelihood is 80% because low security or reputation still offers some reliability for the buyer. For the same reason, we choose 20% as the lowest likelihood because there can not be complete reliability or security. We can change these values for different implementations and applications.

We model the leakage profile $PRV^i_{lea}(t_n)$ of B_i as a tuple given by:

$$PRV^i_{lea}(t_n) =< I^i(t_n), L^i(t_n), \{I^i(t_1), I^i(t_2), ..., I^i(t_l)\} > \tag{4.22}$$

where $I^i(t_n)$ and $L^i(t_n)$ are used in risk assessment based on the potential impact and likelihood that a data leak will occur. $I^i(t)$ is the damage caused by the data leak incident in the past at time t where $t < t_n$.

To quantify the potential impact ($I^i(t_n)$), we are considering the comprehensive purchase history of buyer B_i, i.e., data bought from all providers so far. We compute the following impact attributes: the volume ($vol(t_n)$) and sensitivity ($sen(t_n)$) based on their overall data possession for all the providers; the number of agreements ($agr(t_n)$) they has formed that might get violated if a data leak occurs; and the number of providers ($pro(t_n)$) whose data they has purchased. Suppose B_i bought data from N_P number of providers in the time period $t = [t^i_0, t_n]$. they made N_{DP_j} number of agreements with DP_j, $\forall j = [1, N_P]$. We evaluate these attributes at time t_n using Equation 4.23-4.25.

$$vol(t_n) = \sum_{j=1}^{N_P} \sum_{t=t^i_0}^{t_n} a^{ij}_5(t) \tag{4.23}$$

$$sen(t_n) = \frac{1}{N_P} \sum_{j=1}^{N_P} \frac{\sum_{t=t_0}^{t_n} (a^{ij}_1(t) + a^{ij}_2(t) + a^{ij}_3(t))}{3N_{DP_j}} \tag{4.24}$$

$$agr(t_n) = \sum_{j=1}^{N_P} N_{DP_j} \tag{4.25}$$

$$pro(t_n) = N_P \tag{4.26}$$

where $a^{ij}_5(t)$ corresponds to the sample size, $a^{ij}_1(t)$, $a^{ij}_2(t)$, $a^{ij}_3(t)$ represents age, interval and period respectively. We explained these privacy elements in section 4.3.2. Scores are assigned to the obtained impact attributes using Fig. 4.9, which are then used to calculate $I^i(t_n)$ as per Equation 4.17.

Besides potential damage due to the overall data possession of the buyer, we also consider the impact of any potential data leak events experienced by the buyer in modelling the leakage profile. Recall from section 4.2, when a data leak event occurs, a regulator investigates the data leak to determine the impact. they submits an investigation report to the KYBChain comprising the volume and sensitivity of leaked data, the number of agreements violated and the number of providers affected. KYBChain assign scores to these attributes as per Fig. 4.9 and using Equation 4.17 obtain $I^i(t)$ for the data leak incident that occurred at time t where $t < t_n$.

We adopt the logistic function to model the leakage rating given by Equation 4.27.

$$PR^i_{lea}(t_n) = \frac{1}{1 + e^{-5R^i(t_n)}} \tag{4.27}$$

where

$$R^i(t_n) = I^i(t_n) \times L^i(t_n) \tag{4.28}$$

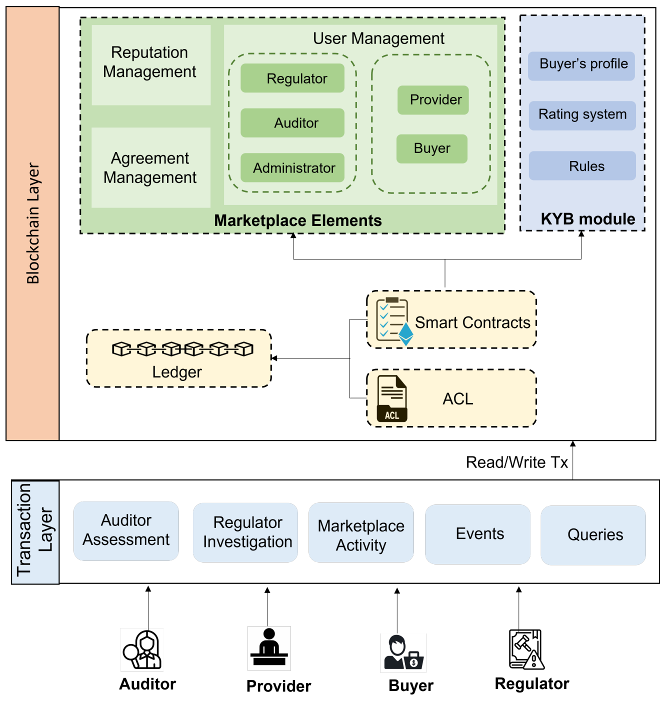
$R^i(t_n)$ is the risk assessment calculated based on the potential impact associated with the overall current possession of the buyer and the likelihood of a data leak event.

4.4 Data marketplace with privacy rating system

In this section, we first present KYBChain architecture that integrates the proposed privacy rating system as discussed in section 4.3 with the decentralized marketplace. Second, we explain the different phases in KYBChain to update and retrieve a buyer's different profiles.

4.4.1 KYBChain Architecture

As depicted in Fig. 4.10, KYBChain architecture is organized into two layers: Transaction (Tx) and Blockchain (BC) layers. The Tx layer encompasses different read/write transactions and events. These transactions are issued by participants that invoke the smart contracts to autonomously execute trade-related activities and update buyers' profiles. Events are emitted on predefined conditions to notify the participant of other participants' activities. At the BC layer, the transactions are processed following a set of access



t!]

Figure 4.10: KYBChain architecture

rules defined by the Access Control List (ACL), which are decided by the network administrator during bootstrapping. The transactions invoke the smart contracts in the BC layer that autonomously execute trade-related activities and automatically map buyers' activities to their different profiles.

We implement KYBChain on a permissionless blockchain network. It is managed by a network administrator who has administrative control over the blockchain and defines the network model. they is also responsible for adding a regional regulator to the network. We choose Ethereum for the deployment of KYBChain. The proposed design is blockchain platform agnostic as it does not use any feature of the consensus or the communication layer and hence can be implemented on most generic blockchain platforms, as long as it provides Turing-complete programming capabilities. We assume that each participant maintains a changeable public/private key pair to be identified in the network.

In the following sections, we describe the layers of the KYBChain framework in detail.

4.4.1.1 Transaction Layer

The Tx layer has inputs from auditors, regulators, buyers, and sellers. These inputs generate transactions to execute the smart contracts in the BC layer. We present different transactions that are categorized based on which participant issued them and their details as follows:

Trading-related activities: The buyer or the seller initiates all the marketplace trading-related transactions. Marketplace end-users issue $Tx_{registerUser}$ to register themselves as a buyer, provider, dual role or auditor handled by user management. $Tx_{registerUser}$ is given by:

$$Tx_{registerU} = [role|Sig_U|PU_U] \tag{4.29}$$

where $role \in$ {buyer, provider, dual or auditor}, Sig_U and PU_U are the user's signature and the public key. Following registration, the buyer's profile is created in the KYB-module using $Tx_{createProfile}$, which is bound to a spawned contract instance from a *BuyerSc* template. A *BuyerSc* manages and maintains the buyer's practice, purchase and leakage profiles.

Newly-registered buyers are initially marked as KYB-unapproved. Such buyers do not receive any rating. Hence, providers may be reluctant to trade with them. To be KYB-approved, a buyer must upload their privacy policy and audit report in a decentralized peer-to-peer blockchain-compatible repository, IPFS (InterPlanetary File System). they requires to record its content id (pointer on IPFS to access the privacy policy) on BC using $Tx_{uploaDPolicy}$ given by:

$$Tx_{uploadPolicy} = [H_{data}|CID|Sig_B|PU_B] \tag{4.30}$$

where H_{data} is the hash of buyer's data stored in IPFS, CID is the content ID to locate their files on IPFS, Sig_B and PU_B are the signatures and public key of the buyer.

All the other trading-related transactions vocabulary is adopted from MartChain (see Chapter 3). These transactions include $Tx_{registerA}$ for registering new data subscription contracts (*SubscriptionSc*) in the KYBChain based on the terms decided between buyer and provider; Tx_{addS} for adding agreed upon specifications of a new subscription in their *SubscriptionSc*. Data is transferred off-chain from provider to buyer. After which, both confirm the data delivery and share their feedback based on their experience via transaction $Tx_{confirmDelivery}$. This transaction performs settlement by transferring payment from the buyer's escrow account to the provider's account, marking the data subscription completed. Besides, it also triggers the update of the buyer's latest purchase as per the specification of the data subscription in their purchase profile.

Auditor assessment: The buyer's $Tx_{uploadPolicy}$ emits an event to the set of registered auditors who are randomly selected using the hash-sharding method [225] as used in Distributed SQL Database. This method evenly segments the list of registered auditors into multiple shards, assigning a unique shard ID to each. The hash of the transaction ID of $Tx_{uploadPolicy}$ is used to identify and select a random shard. The audit request is sent to all the auditors in the selected shard. With a uniform hashing algorithm such as Keccak-256, the hash function can evenly distribute the audit request across different shards of auditors, reducing the risk of repeatedly selecting the same shard. Recall from section 4.3.1 randomizing the auditor selection prevents any chance of collusion between buyer and auditor that can manipulate assessment. Selected auditors then retrieve the content ID from the KYBChain and the files from the IPFS. Auditors assess the buyer's policy and audit report and submit their assessment report using $Tx_{auditReport}$. The buyer's status is changed to KYB-approved when 2/3 of the auditors in shards submit positive assessment results. This will ensure better system security than a pure majority (51%). Moreover, the final score of each privacy element is determined by averaging the corresponding score submitted by all the auditors. However, if the 2/3 condition is not met, the buyer's status remains KYB-unapproved. Such buyers may require to improve the efficacy of their privacy practice per the regulatory requirements and request re-assessment. $Tx_{auditReport}$ is given by Equation 4.31.

$$Tx_{auditReport} = [PU_{B_i}|Result|\text{Assessment report}|t_{audit}|Sig_A|PU_A] \tag{4.31}$$

where *Result* is either pass or fail based on whether or not B_i complies with the data protection law as per the auditor's assessment. *Assessment report* contains scores (x_1^i, x_2^i,

x_3^i, x_4^i, x_5^i, x_6^i) given to each privacy element of the practice profile (see section 4.3.1). Sig_A and PU_A are the signatures and public keys of the auditor. t_audit is when the audit report was submitted and is used to determine $T_{threshold}$ in Equation 4.6.

Regulator investigation: The role of the regulator is significant when a data leak event occurs. A regulator can discover a suspected data leak event directly or can be alerted by some external sources. The regulator identifies the buyer responsible for the leak, collects evidence, and investigates the data leak event. Based on the analysis, they issues $Tx_{investgationReport}$ to submit their investigation report given by

$$Tx_{investgationReport} = [PU_{B_i}|\text{Investigation report}|Sig_R|PU_R] \tag{4.32}$$

where PU_{B_i} is the public key of the identified buyer B_i. *Investigation report* contains the following details: t_leak, list of affected providers, leaked data sensitivity, agreements violated, and the volume of leaked data>. t_leak is the time when the leak was detected for a buyer. Sig_R and PU_R are the regulator's signature and public key, respectively. This transaction updates the buyer's leakage profile. Besides, it also emits an event to all the affected providers to notify them about the leak per the Notifiable Data Breach (NDB) scheme of Privacy Act [203].

KYBChain uses several internal transactions that connect marketplace elements with the KYB-module. These internal transactions are issued by *RegisterSc* to *BuyerSc* triggered by the participants' transactions. $Tx_{updatePractice}$ update auditor's privacy assessment in the practice profile triggered by $Tx_{auditReport}$, $Tx_{updateImpact}$ update the regulator's investigation in the leakage profile triggered by $Tx_{investgationReport}$ and $Tx_{updatePurchase}$ update subscription detail in the purchase profile triggered by $Tx_{confirmDelivery}$. These transactions are given by:

$$Tx_{updatePractice} = [PU_{B_i}|Assessment\ report|t_{audit}] \tag{4.33}$$

$$Tx_{updateImpact} = [PU_{B_i}|t_{leak}|\text{Investigation report}] \tag{4.34}$$

$$Tx_{updatePurchase} = [PU_{B_i}|PU_{DP}|t_{purchase}|\text{Subscription detail}] \tag{4.35}$$

where *Subscription detail* consists of type, age, interval, period, resell consent, and retention. $t_{purchase}$ is when the subscription is finished. DP is the provider whose data the buyer has purchased.

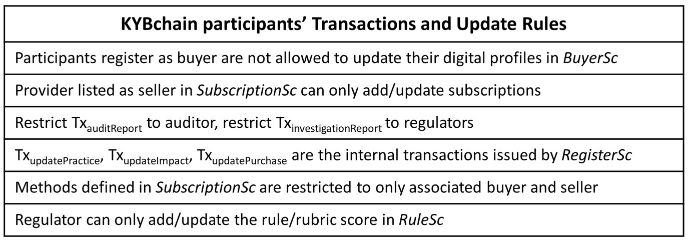
Query: Participants issue $Tx_{retrieveProfiles}$ to retrieve the buyer's profile which is given by:

$$Tx_{retrieveProfiles} = [\text{Option} \mid \text{Preference} \mid data | PU_{B_i} | PU_U | Sig_U] \tag{4.36}$$

A participant can choose from two *option* to obtain a buyer's profile: (i) Coarse: selecting this option, $Tx_{retrieveProfiles}$ returns coarse-grained privacy profile, i.e., $PRV^{ij} = < PR^i(t),\ PR^i_{pra}(t),\ PR^{ij}_{pur},\ PR^i_{lea} >$, (ii) Fine: this option return fine-grained privacy profiles, i.e., $< PRV^i_{pra}(t),\ PRV^{ij}_{pur},\ PRV^i_{lea} >$. These components are calculated as explained in section 4.3. The privacy rating of buyers serves different purposes for different entities. Regulators and buyers are more interested in fine-grained profiles to ensure privacy compliance. In contrast, a provider can use a coarse option to assess the risk of sharing data with a particular buyer per their demand. *preference* and *data* are empty in the case of regulator and buyer, while for the provider, it is set to their weight preference and purchase requests for which the provider wants to measure their exposure level.

Events: Several events are emitted that asynchronously notify participants about different activities happening on the KYBChain. These are listed below.

- *subscriptionAdded*: This is the trade event by *SubscriptionSc* to notify the buyer that a seller has added a subscription to their contracts based on their negotiated terms. A buyer can verify the subscription details and raise a dispute if anything is wrong.
- *auditRequest*: This event is emitted by *RegisterSc* when the buyer updates their IPFS content ID to inform the registered auditors of the pending assessment. This event contains details of the buyer requesting a privacy assessment.
- *auditResult*: This event is emitted by *RegisterSc* to inform the buyer about the outcome of their privacy assessment.



KYBchain participants' Transactions and Update Rules
Participants register as buyer are not allowed to update their digital profiles in *BuyerSc*
Provider listed as seller in *SubscriptionSc* can only add/update subscriptions
Restrict $\mathrm{Tx}_{\mathrm{auditReport}}$ to auditor, restrict $\mathrm{Tx}_{\mathrm{investigationReport}}$ to regulators
$\mathrm{Tx}_{\mathrm{updatePractice}}$, $\mathrm{Tx}_{\mathrm{updateImpact}}$, $\mathrm{Tx}_{\mathrm{updatePurchase}}$ are the internal transactions issued by *RegisterSc*
Methods defined in *SubscriptionSc* are restricted to only associated buyer and seller
Regulator can only add/update the rule/rubric score in *RuleSc*

Figure 4.11: ACL defines the rules of access to the resources of the KYBChain

- *leakNotification*: This event is triggered when the regulator submits their investigation report of a data leakage incident. All the providers who are impacted by the data leakage are notified of the incident by *RegisterSc*.

4.4.1.2 Blockchain Layer

Transactions in KYBChain are governed by an ACL which defines the permissions for submitting transactions, read/write access to the ledger, updating profiles, and other actions for the participants as shown in Fig. 4.11. The BC layer employs two sets of smart contracts for marketplace elements and the KYB-module. Marketplace elements include the user's registration for all marketplace entities, trading agreements between buyer and seller and their respective reputation score. All the users who register as buyers will be mapped to their digital profile using their public key in the KYB-module. ACL does not permit buyers to modify their digital profiles. Next, smart contracts are invoked by transactions received from the Tx layer to automatically update and evaluate privacy ratings. The following Sections present the smart contracts of the BC layer in detail:

Marketplace elements: KYBChain is the extension MartChain (see Chapter 3) from which we adopted the marketplace elements, including user management, agreement management and reputation management. These marketplace elements are implemented using the following smart contracts: *RegisterSc*, *SubscriptionSc*, *ReputationSc*, which are installed by the administrators during bootstrapping.

RegisterSc provides core marketplace functionalities such as entity registration and agreement management. *registerUser()* allows participants to register themselves. Registered users can deploy new data subscription contracts and register them using *registerAgreement()*. Besides, *RegisterSc* oversees all the trade-related activities for the registered contracts. Moreover, *RegisterSc* also manages pertinent information, such as the details of the registered auditors, regulators and administrators. To integrate the KYB-module with the marketplace, we extended the *RegisterSc* design by adding various interfaces. Auditors who are interested in enrolling in KYBChain, register themself using *registerAuditor()*. Regulators are added by the administrator using *addRegulator()*. Auditor uses *auditReport()*, and regulator uses *investgationReport()* to submit their audit report and leakage investigation report, respectively. Another method *uploadPolicy()* is provided for the buyer to update their IPFS content ID. *retrieveProfile()* to retrieve the privacy rating of the buyer. These newly added functions are designed with role-based permissions as given by Fig. 4.11 to restrict other entities from unauthorized access.

SubscriptionSc template smart contract allows participants to customize it based on their terms and conditions. When a buyer and seller agree to trade with each other, a new instance of *SubscriptionSc* is spawned and deployed in the KYBChain. *SubscriptionSc* is a bilateral contract that binds buyer-seller and ensures that their behaviour automatically conforms to the terms. Furthermore, *SubscriptionSc* maintains the list of data subscriptions and automatically manages their execution status using several methods: *addAgreement()* to add new data subscription, *startSubscription()* to commence the data transfer and *confirmDelivery()* to confirm the transfer of data to initiate the payment and settlement. User also submits their feedback based on their trade experience. The feedback is used to evaluate the reputation score of participants in the marketplace that is managed and maintained by *ReputationSc*. It is important to note that the reputation score of the buyer differs from their privacy rating as the former is based on the trading experience of sellers with him while the latter solely depends on the buyer's profiles that are handled by the KYB-module, as explained next.

KYB-module: KYB-module consists of three smart contracts: *PrivacyRatingSc*, *RuleSc*

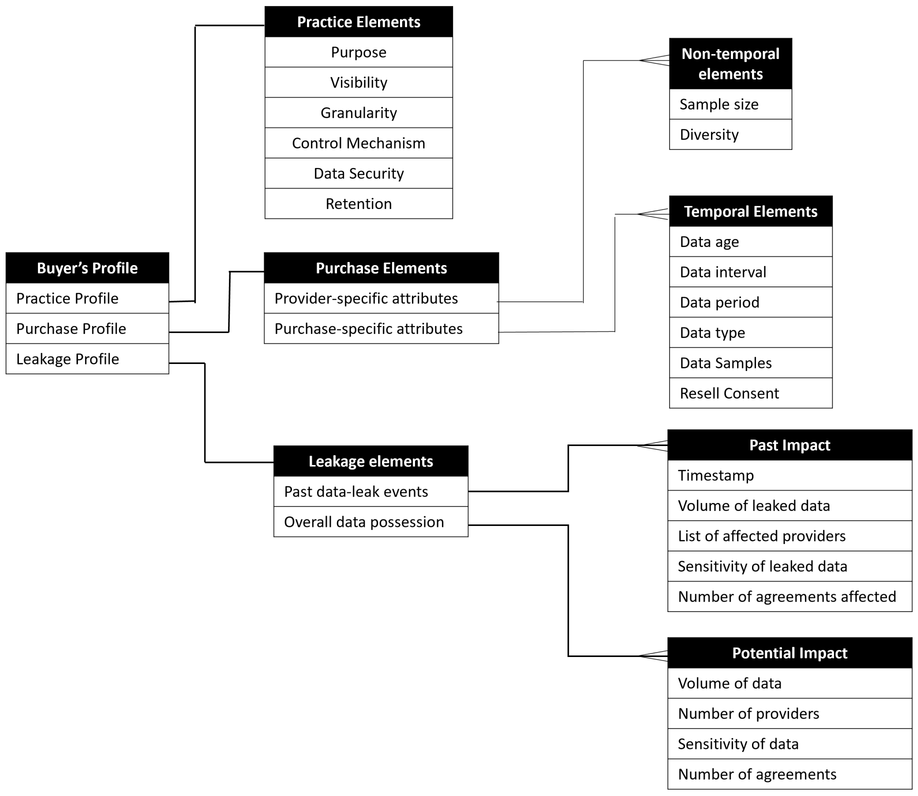


Figure 4.12: Entity relation diagram illustrating a data structure for storing buyer's profile

and *BuyerSc*. The logic to update, retrieve and compute privacy profiles is implemented in *PrivacyRatingSc*, and the data structure to record a buyer's profiles is implemented in *BuyerSc*. Decoupling logic from data storage facilitates the upgradeability of the privacy rating evaluation system without affecting the buyer's profile [226]. *PrivacyRatingSc* connects the KYB-module with the marketplace elements via *RegisterSc*. For any related buyer's activities in the marketplace, *RegisterSc* sends transactions to *PrivacyRatingSc* to update buyers' profiles in their associated *BuyerSc*. *PrivacyRatingSc* also implements logic to retrieve and compute privacy profiles using the rules specified in *RuleSc*. *RuleSc* is a smart contract that defines all the rubric scoring tables and rules for grading systems as explained in section 4.3. *RuleSc* is deployed in the network by the regulators with read-only access that can only be upgraded by deploying a new contract. *PrivacyRatingSc* manages all the *BuyerSc* that collect and store data about buyers' profiles.

We design *BuyerSc* as a template smart contract comprising data structures and functions to record buyers' profiles in the blockchain. When a participant registers himself as a buyer, a new *BuyerSc* is spawned from the template smart contract. The *BuyerSc* address is registered in *RegisterSc* against the buyer's identifier. The *BuyerSc* records buyer's practice, purchase and leakage profiles in a data structure that is implemented based on the entity-relationship diagram as shown in Fig. 4.12. The privacy practice profile consists of all the privacy elements defined in section 4.3.1 and is initialized based on the privacy audit report submitted by the auditors. Next, a separate purchase profile is maintained for each provider whose data the buyer had purchased. The purchase profile is designed as a nested structure that consists of a provider-specific structure and an array of data-specific structures. The provider-specific structure monitors the non-temporal elements depending on the overall purchases of a specific provider. It includes the total number of data samples and a list of data types. The array of data-specific structures records temporal elements corresponding to each subscription for a particular provider. It includes age, interval, and period. Lastly, the leakage profile comprises privacy elements discussed in section 4.3.3 that are initialized based on the investigation report submitted by the regulator. Depending on different events and activities, the corresponding profile of the buyer is updated, and the privacy rating is dynamically evaluated.

4.4.2 KYBChain Phases

This section presents the details of the various phases of KYBChain, including setup and initialization, profiles update in different stages, and retrieval.

4.4.2.1 Setup and initialization

During this phase, the KYBChain network is bootstrapped. The administrator sets up KYBChain by deploying smart contracts of the KYB-module and integrating it with the marketplace element via *RegisterSc*. As discussed earlier, *RegisterSc* is extended with new functions and transactions such as $Tx_{auditReport}$, $Tx_{investgationReport}$, $Tx_{uploaDPolicy}$, $Tx_{createProfile}$ etc., that are required by different participants to interact with the KY-

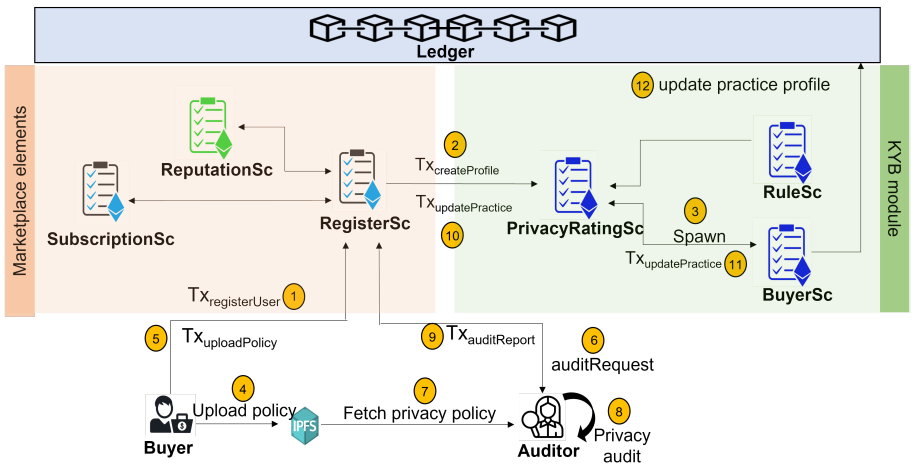


Figure 4.13: Initial auditing Stage

BChain. There are two ways to integrate the KYB-module with an existing marketplace. (i) a new version of *RegisterSc* integrated with new functions can be deployed, and all states from the old contract are migrated to the new one. (ii) existing *RegisterSc* can be upgraded by using update plugin of OpenZeppelin [227]. During the initialization phase, regulators are added to the network by the administrator, and they deploy *RuleSc* in the KYB-module. External auditors are initially incentivized to register in KYBChain through an initial coin offering (ICO) and receive native tokens for their audit assessment service.

4.4.2.2 Update of buyer's profile

In this section, we present end-to-end interactions of smart contracts during different stages of KYBChain that trigger the update of the buyer's profile.

Initial auditing : During this stage, the privacy practice of the buyer is assessed and updated in their practice profile. The end-to-end interactions of smart contracts during this stage are depicted in Fig. 4.13. In Step 1, a user register himself by issuing $Tx_{registerUser}$ to *RegisterSc* and receive a dynamic key pair $PK+/PK-$. If the user has selected buyer as a role, *RegisterSc* subsequently sends $Tx_{createProfile}$ in Step 2 to *PrivacyRatingSc* that

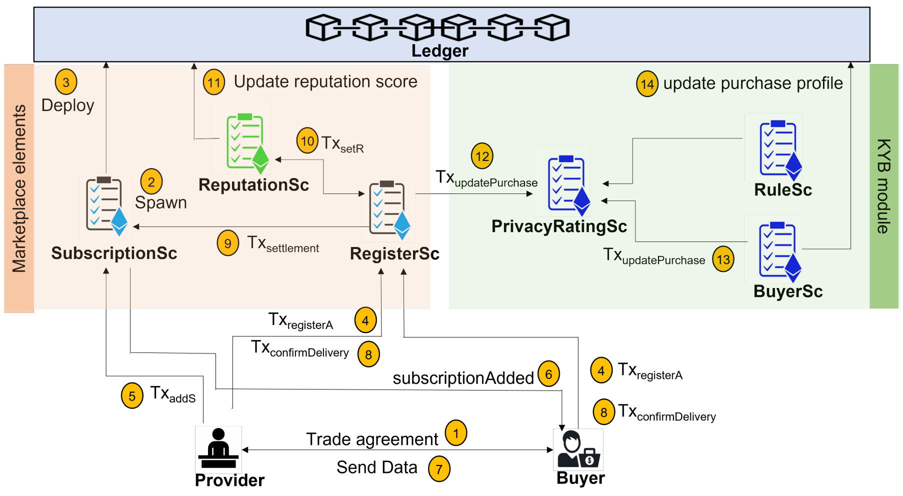


Figure 4.14: Pre-assessment Stage

spawns a new instance of *BuyerSc* in Step 3. The contract address is recorded against the buyer's identifier in their user profile maintained by *RegisterSc*. In Step 4, the buyer uploads their privacy policy and audit report on an IPFS. In Step 5, the buyer issues $Tx_{uploadPolicy}$ to *RegisterSc* to share the content ID. In Step 6, *RegisterSc* emits an event *auditRequest* to the set of randomly selected registered auditors. In Steps 7-8, auditors fetches the buyer's privacy policy and audit report, perform an assessment, and submit their report using $Tx_{auditReport}$ in Step 9. A positive assessment changes the buyer's status to KYB-approved, which issues $Tx_{updatePractice}$ to the *PrivacyRatingSc* in Step 10 that forwards the transaction to the corresponding *BuyerSc* in Step 11. The final scores are updated in the practice profile of the buyer in Step 12.

Pre-assessment: During this stage, the buyer's purchases are monitored and recorded in their purchase profile as discussed in section 4.3.2. The end-to-end interactions of smart contracts are depicted in Fig. 4.14. In Step 1, a buyer and provider agree to trade with each other, and an instance of *SubscriptionSc* contract is spawned and deployed in Step 2-3. Both provider and buyer issue $Tx_{registerA}$ to *RegisterSc* to register the contract address of *SubscriptionSc* in Step 4. Based on the buyer's data requirement, the provider adds a new subscription in the *SubscriptionSc* in Step 5, about which the buyer is notified via an event *subscriptionAdded* in Step 6. The subscription status is set to ACTIVE when the provider

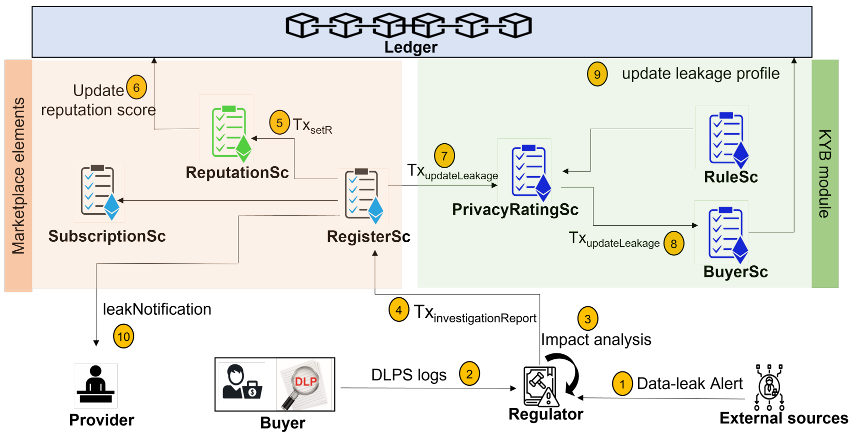


Figure 4.15: Post-management Stage

transfers the data off-chain over a secured channel to the buyer in Step 7. After the transfer, both parties confirm the data delivery by issuing $Tx_{confirmDelivery}$ to *RegisterSc* in Step 8. This triggers the settlement process, during which payment is transferred from the buyer's account to the provider's account. *RegisterSc* sends three transactions to different contracts. In Step 9, $Tx_{settlement}$ is issued to *SubscriptionSc*, which changes the subscription status to FINISH. In Steps 10-11, Tx_{setR} is sent to *ReputationSc*, which maintains the reputation score of each user (see Chapter 3). Both parties submit feedback based on their trading experience, which is utilized to evaluate their respective reputation score. In Steps 12-14, *RegisterSc* issue $Tx_{updatePurchase}$ to *PrivacyRatingSc* that forwards the transaction to the corresponding *BuyerSc* to update the buyer's purchase profile.

Post-management: During this stage, the buyer's leakage profile is updated if any data leak event occurs. The end-to-end interactions of smart contracts are depicted in Fig. 4.15. In Step 1, the regulator becomes aware of the data leak events either via self-discovery or alert from external sources [228] such as the buyer himself or the service provider etc. In Step 2, the regulator gathers evidence from the guilty buyer, e.g. logs from their data leakage prevention system [229] to investigate the cause and impact of leakage in Step 3. In Step 4, they submits their investigation report to *RegisterSc* using $Tx_{investigationReport}$. $Tx_{investigationReport}$ leads to three actions by *RegisterSc*. Firstly, Tx_{setR} is issued to *ReputationSc* to update the buyer's reputation score in Steps 5-6 to

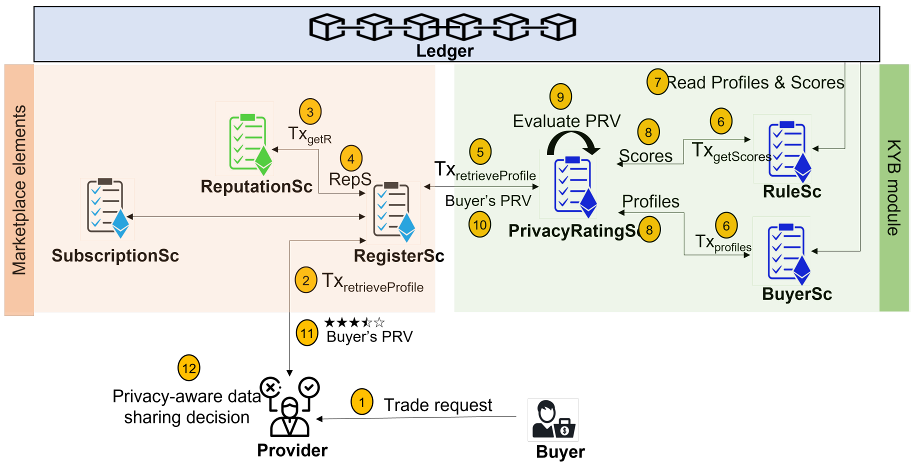


Figure 4.16: Retrieval of buyer's privacy profile

reflect their reputation damage. Secondly, $Tx_{updateLeakage}$ is sent to *PrivacyRatingSc* in Steps 7-9, which is forwarded to the corresponding *BuyerSc* of the concerned buyer. This is to update the buyer's leakage profile per the investigation report submitted by the regulator. Lastly, *RegisterSc* emits an event *leakNotification* to all the affected providers to notify them of the data leak event in Step 10.

4.4.2.3 Retrieval of buyer's privacy profile

The end-to-end smart contract interactions to retrieve the buyer's privacy profile are depicted in Fig. 4.16. On receiving the purchase request of a buyer in Step 1, the provider sends a query $Tx_{retrieveProfile}$ to textitRegisterSc to fetch the buyer's profiles in Step 2. Provider shares their privacy preferences and purchase request sent by the buyer in the transaction. In Step 3-4, textitRegisterSc fetches the reputation score ($RepS$) of the buyer using Tx_{getR} and forwards the retrieval request to the *PrivacyRatingSc* in Step 5. In Steps 6-8, *PrivacyRatingSc* executes various read transactions to retrieve the practice profile, current possession and future acquisition specific to the provider who initiated the request and leakage profile from the *BuyerSc*. Subsequently, it retrieves rubric score guides using the *RuleSc*. In Step 9, *PrivacyRatingSc* compute the buyer's privacy profile (PRV) using all this retrieved information and the provider's preferences. In Step 10, *PrivacyRatingSc* forwards the buyer's obtained *PRV* to *RegisterSc* that forwards it to the provider in Step 11. In Step 12, using the privacy rating vector of the buyer, a provider could act as per their privacy preferences and take three possible decisions based on the associated privacy risk of data sharing with the buyer: (1) if the risk is low, the provider can accept the purchase request or 2) if the risk is medium to high, they can negotiate the price and terms with the buyer or 3) if the risk is too high, they can decide to reject the purchase request.

4.5 Evaluation

The purpose of this evaluation section is two-fold: (1) to illustrate the efficacy and practical utility of privacy rating and (2) to conduct a quantitative performance evaluation of the KYBChain architecture. We conduct a comparative analysis of the proposed marketplace system integrated with privacy rating, termed "with rating", with the baseline marketplace system where no privacy rating is used, termed as "without rating". We performed all the experiments and performance tests on a 3.70GHz Intel(R) Xeon(R) 12 Linux Server (Ubuntu) with 62GB RAM Memory.

4.5.1 Utility of the privacy rating

In this section, we first describe our simulation setup, simulation methodology, performance metrics, and experiment results to demonstrate the utility of privacy rating.

4.5.1.1 Simulation Set-up

Since there is no commercially available data for the marketplace participants, we employed synthetic datasets by generating an (i) sample space of buyers with different profiles and (ii) sample space of providers with different behaviours characterized based on their privacy attitude.

Buyers' Sample space: In our simulation, we have considered N_B number of buyers in the marketplace. We generate each buyer's (B^i) profile by randomly choosing scores for the corresponding privacy elements using a uniform distribution. The choice of uniform distribution suits our scenario since it defines equal probability for each risk score in the given set. We generate these profiles in the time period (t_0^i, t_n) where t_0^i and t_n correspond to the time B_i starts trading and t_n is the time when we evaluate their privacy rating, respectively. Different parameters used to simulate buyers' different profiles are explained below:

- Practice profile: Recall from section 4.3.1, the purchase profile consists of the score of privacy elements $(x_1^i, x_2^i, x_3^i, x_4^i, x_5^i, x_6^i)$ which we uniformly select from the discrete set $[-1, -0.5, 0, 0.5, 1]$. We assume $T^i_{threshold}$ has not expired $(t_n \leq T^i_{threshold})$, which means the audit assessments are effective for each B_i in the time period of interest.
- Purchase profile: To calculate the purchase rating $(PR^i_{pur}(t_n))$, purchase history for each B_i is built. We considered the maximum number of data subscriptions formed by all the buyers to be X_{max}. We randomly select the number of data subscriptions for each B_i from $[1, X_{max}]$. Each data subscription of B_i is with a provider DP_j randomly selected from the set $\{DP_U, DP_P, DP_F\}$ as explained later. For each purchase of B_i to buy DP_j's data at time t, the risk scores for temporal ($a_1^{ij}(t)$, $a_2^{ij}(t)$, $a_3^{ij}(t)$) and non-temporal ($a_4^{ij}(t)$, $a_5^{ij}(t)$) privacy elements are selected from the discrete set $[-1, -0.5, 0, 0.5, 1]$ (see section 4.3.2) following a uniform distribution.
- Leakage profile: The privacy elements for leakage profile, i.e. $sen^i(t)$, $vol^i(t)$, $agr^i(t)$, $pro^i(t)$ to evaluate the potential impact $(I^i(t))$ of B^i, are calculated using their generated purchase history. To calculate likelihood $(Li^i(t))$ of data leak for each B^i, reputation score $(RepS^i_{DL}(t))$ is randomly selected using uniform distribution between $[0, 1]$. Recall from section 4.3.3 the losses incurred by the historical data leak events are used to determine the buyer's reputation score. Since we have randomly selected the buyer's reputation score, the effect of I^i_{past} is assumed to be already considered in the analysis.

At the current time t_n, we consider that each B_i query the marketplace by sending two purchase requests (i) tr^i_{LCD}: a demand to request a low critical data that relatively poses a low privacy risk to the provider. The risk score of each privacy element for tr^i_{LCD} is uniformly selected in $\{-1, -0.5, 0\}$ implying low risk and (ii) tr^i_{HCD}: a demand to trade high critical data that poses relatively high privacy risk to the provider. The risk score of each privacy element for tr^i_{HCD} is uniformly selected in $\{0, 0.5, 1\}$ implying high risk. Each purchase request consists of the buyer's data requirements (d, $a^i_1(t_n)$, $a^i_2(t_n)$, $a^i_3(t_n)$) and their willingness to pay (WTP) amount, i.e., the maximum price buyer is willing to pay to procure the data as per their requirement [147]. We assume that the buyer's WTP amount depends on the number of data samples and the criticality of the requested data. "Willingness to pay" of tr^i_{LCD} and tr^i_{HCD} are represented by WTP^i_{LCD} and WTP^i_{HCD} respectively and given by Equation 4.37 and 4.38.

$$WTP^i_{LCD} \sim \begin{cases} \mathcal{U}[1, \frac{LCD_{max}}{2}] & \text{if } a^i_5(t_n) < 0 \\ \mathcal{U}[1 + \frac{LCD_{max}}{2}, LCD_{max}] & \text{if } a^i_5(t_n) \geq 0 \end{cases} \tag{4.37}$$

$$WTP^i_{HCD} \sim \begin{cases} \mathcal{U}[LCD_{max} + 1, \frac{HCD_{max}}{2}], & \text{if } a^i_5(t_n) < 0 \\ \mathcal{U}[1 + \frac{HCD_{max}}{2}, HCD_{max}], & \text{if } a^i_5(t_n) \geq 0 \end{cases} \tag{4.38}$$

where LCD_{max} and HCD_{max} are the maximum WTP amount for all the LCD and HCD purchase requests, respectively.

Providers' Sample space: As presented in [230], individuals have different privacy beliefs, concerns, and attitudes towards data sharing. Moreover, to design a privacy-aware marketplace system, understanding providers' preferences is important [85]. In this section, we define providers' preferences based on their different privacy characteristics to generate the providers' sample space.

Provider's preference in the data marketplace expresses their wishes about accepting buyers' purchase requests and sharing their data with them. Since a provider's motive to trade their data is economically driven; therefore, we consider their "willingness to accept" (WTA) amount in their preference. Willingness to accept is the minimum monetary amount that a provider is willing to accept to trade their data [147]. Moreover, a privacy-concerned provider would also have a privacy preference that we define in terms of privacy rating. We represent a provider's (DP_j) preference by a tuple given by:

$$Pref^j = < PR^j, PR^j_{pra}, PR^j_{pur}, PR^j_{lea}, WTA^j > \tag{4.39}$$

where $Pref^j$ is DP_j's preference. PR^j, PR^j_{pra}, PR^j_{pur}, PR^j_{lea} and WTA^j are their preferences for privacy rating, practice rating, purchase rating, leakage rating and willingness to accept, respectively.

We sought to investigate how different providers benefit from using privacy ratings in managing the risk-utility trade-off [230] posed by trading their sensitive IoT data in return for monetary benefits. To this end, we employed Westin's privacy segmentation [231] of individuals based on their privacy attitude profiles. In our analysis, we considered three providers with different privacy characteristics: an unconcerned privacy provider (no or low privacy concern), a privacy pragmatist provider (mid-level privacy concern), and a privacy fundamentalist provider (high privacy concern). We describe the privacy characteristics of these providers and their preferences as follows:

- Unconcerned provider (DP_U): According to [232], about 20% of individuals can be categorised as privacy unconcerned. These individuals are comfortable sharing their data and believe such behaviour does not threaten their privacy. Therefore, we assume that the privacy-unconcerned provider is generally willing to trade their data to buyers, regardless of the threat to their privacy. Therefore, we define their preferences only in terms of price since privacy preferences seem inconsequential to him. It is given by Equation 4.40.

$$Pref^U = \begin{cases} \{0, 0, 0, 0, WTA^U\} & \text{in "w/o rating"} \\ \{0, 0, 0, 0, WTA^U\} & \text{in "with rating"} \end{cases} \tag{4.40}$$

  where $Pref^U$ is DP_U preference. WTA_U is the price preference. As per our assumption, their privacy preference is set to zero, which means that an unconcerned provider will accept most of the purchase requests they receives solely based on their price preference, irrespective of the sensitivity of the requested data.

- Fundamentalist provider(DP_F): According to [233], privacy fundamentalists account for about 25% of the population. These individuals usually have little interest in providing their data. Since a fundamentalist provider willingly registers in the marketplace for monetary benefits, we assume they would be very selective in accepting purchase requests and conservatively trade their data. We make the following assumptions about the fundamentalist provider: (i) without a rating system, the fundamentalist provider would read the buyers' privacy policy to understand their privacy practice before sharing their data [233], (ii) with a rating system, they would consider the fine-grained components of privacy rating vector in decision-making, (iii) they would be restrictive in accepting highly sensitive data requests and hence, their privacy preferences and price preferences for HCD requests would be stringent than LCD requests. Fundamentalist providers' preferences for both marketplace settings are given by Equation 4.41.

$$Pref^U = \begin{cases} \{0, PR^F_{pra}, 0, 0, WTA^F\} & \text{in ``w/o rating''} \\ \{0, PR^F_{pra}, PR^F_{pur}, PR^F_{lea}, WTA^F\} & \text{in ``with rating''} \end{cases} \tag{4.41}$$

where $Pref^F$ is DP_F preference. Based on our assumption, DP_F prefers the practice rating (PR^F_{pra}) and is willing to accept it in "w/o rating" setting. While in the case of "with rating", they considers fine-grained components of privacy rating, i.e., practice (PR^F_{pra}), purchase (PR^F_{pur}) and leakage (PR^F_{lea}) rating.

- Pragmatist provider (DP_P): According to [232], privacy pragmatists account for about 55% of the population. They willingly share some personal information for incentives. Pragmatist provider's preference is generally between two extremes, i.e., they is not as lenient as unconcerned and not as stringent as fundamentalist in sharing their data. Based on this argument, we only considered price preference in "w/o rating" and overall privacy rating for their privacy preference in "with rating". DP_P's preference is given by Equation 4.42.

$$Pref^P = \begin{cases} \{0, 0, 0, 0, WTA^P\} & \text{in ``w/o rating''} \\ \{PR^P, 0, 0, 0, WTA^P\} & \text{in ``with rating''} \end{cases} \tag{4.42}$$

Algorithm 3 Algorithm to determine if B_i purchase request is accepted by the DP_j

Buyer's Input: $tr^i(t)$, WTP^i, $PRV^{ij}(t)$, Li^i

Provider's Input: $Pref_j$

Output: $Accepted^j(tr^i)$, $Leaked^j(tr^i)$

```
1: if each element in $Pref_j \geq$ element in $(PRV^{ij}(t), WTP^i)$
2:     $Accepted^j(tr^i) = 1$
3: else
4:     $Accepted^j(tr^i) = 0$
5: Generate random $Pr^i_{leak} \in [0,1]$
6: if $(Pr^i_{leak} < Li^i(t)$ and $Accepted^j(tr^i) == 1)$
7:     $Leaked^j(tr^i) = 1$
8: else
9:     $Leaked^j(tr^i) = 0$
10: return $(Accepted^j(tr^i), Leaked^j(tr^i))$
```

where $Pref^P$ is DP_P preference. WTA^P is their price preference and PR^P is their overall privacy rating preference.

4.5.1.2 Simulation Methodology

To determine the utility of privacy rating, we assume that each $DP_j \in [DP_U, DP_P, DP_F]$ receives purchase requests from N_B buyers in the marketplace. For each purchase request of B_i, we determine whether it is accepted by DP_j or not based on their preference and buyer's privacy rating vector and WTP using Algorithm 3. A purchase request (tr^i) of B_i at time t_n is accepted by DP_j, i.e., $Accepted^j(tr^i) = 1$, if the buyer's privacy rating vector $(PRV^{ij}(t_n))$ and WTP amount (WTP^i) satisfies provider's preference $(Pref_j)$ otherwise it is rejected, i.e., $Accepted^j(tr^i) = 0$. On accepting the purchase request of a buyer, the provider shares their data per the requirement specified in the purchase request. Hence, $Accepted^j(tr^i) = 1$ signifies that the provider (DP_j) has disclosed their data to the buyer (B^i). Next, to observe the utility of leakage rating, we simulate a data leak event for the buyer based on a randomly generated leakage probability (Pr^i_{leak}) and their likelihood of leakage (Li^i). We assume that a buyer (B^i) leaks the provider's disclosed data corresponding to their purchase request tr^i if $Pr^i_{leak} < Li^i(t)$ and $Accepted^j(tr^i) == 1$ are true, i.e., $Leaked^j(tr^i) = 1$ otherwise it is not leaked, i.e., $Leaked^j(tr^i) = 0$

Then, we evaluate the effect of privacy rating usage on the total number of accepted purchase requests by a provider, the provider's revenue generation and the amount of provider's data leaked if a data-leak event occurs. These metrics are explained below:

- **Accepted purchase requests:** This metric refers to the total number of purchase requests accepted by the provider per their preference. This metric is useful to show the impact of using a rating system on the effectiveness of the marketplace, i.e., the number of buyers' demands that are fulfilled by the marketplace.

$$\text{Total accepted purchase requests} = \sum_{i=1}^{N_B} Accepted^j(tr^i) \tag{4.43}$$

- **Revenue:** This metric estimates the provider's total revenue generated as the sum of WTP of all the accepted purchase requests. Comparing this metric in "with rating" and "w/o rating" settings shows the impact of using privacy preference on the provider's revenue generation.

$$\text{Revenue} = \sum_{i=1}^{N_B} \begin{cases} WTP^i & \text{if } Accepted^j(tr^i) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{4.44}$$

- **Leakage percentage:** This metric helps the provider to estimate the percentage of their data leaked of their total disclosed data. Comparing this metric in "with rating" and "w/o rating" settings shows the impact of using leakage rating on the amount of provider' leaked data. A low leakage percentage is preferable since it signifies that a small fraction of data is leaked if data leak events occur for all the buyers, hypothetically, in the marketplace.

$$\text{Leakage percentage} = \frac{\sum_{i=1}^{N_B} Leaked^j(tr^i)}{\sum_{i=1}^{N_B} Accepted^j(tr^i)} \times 100 \tag{4.45}$$

4.5.1.3 Simulations

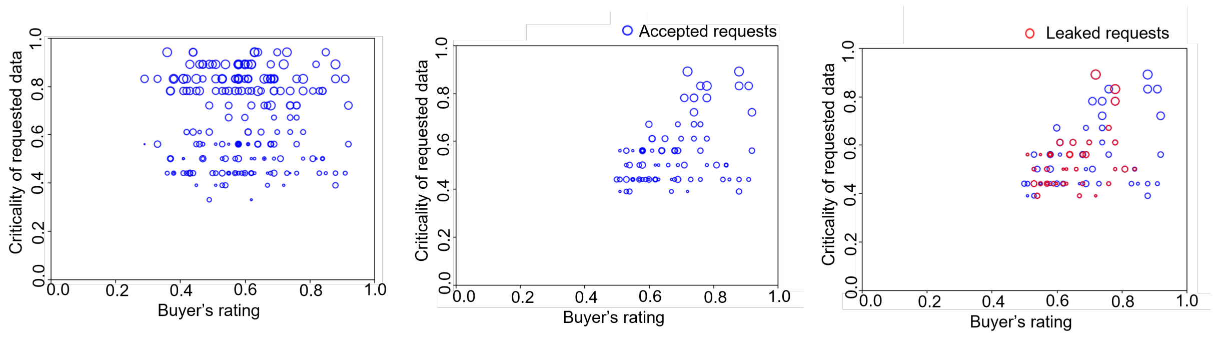
We conduct several simulations to measure performance metrics, as detailed below. For each experiment, we took the average of 10 different runs for statistical significance and also to contemplate all the possible combinations of privacy elements.

Table 4.4: Parameters used in the simulation

Parameter	Value
Total number of buyers	$N_B = 1000$
B_i start time	$t_0^i \sim \mathcal{U}[0,9]$
Current time	$t_n = 10$
Maximum number of trade	$X_{max} = 10$
Maximum WTP for LCD requests	$LCD_{max} = 500$
Maximum WTP for HCD requests	$HCD_{max} = 1000$
Unconcerned preference	$Pref^U = \begin{cases} (0,0,0,0,0) & \text{for LCD requests in “w/o rating”} \\ (0,0,0,0,0) & \text{for LCD requests in “with rating”} \\ (0,0,0,0,0) & \text{for HCD requests in “w/o rating”} \\ (0,0,0,0,0) & \text{for HCD requests in “with rating”} \end{cases}$
Fundamentalist preference	$Pref^F = \begin{cases} (0,0.6,0,0,60\%) & \text{for LCD requests in “w/o rating”} \\ (0,0.6,0.6,0.6,20\%) & \text{for LCD requests in “with rating”} \\ (0,0.8,0,0,80\%) & \text{for HCD requests in “w/o rating”} \\ (0,0.8,0.8,0.8,40\%) & \text{for HCD requests in “with rating”} \end{cases}$
Pragmatist preference	$Pref^P = \begin{cases} (0,0,0,0,50\%) & \text{for LCD requests in “w/o rating”} \\ (0.5,0,0,0,10\%) & \text{for LCD requests in “with rating”} \\ (0,0,0,0,70\%) & \text{for HCD requests in “w/o rating”} \\ (0.7,0,0,0,30\%) & \text{for HCD requests in “with rating”} \end{cases}$
PR weight preferences	$(w_1, w_2, w_3) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$
PES weight preferences	$(\alpha_1, \alpha_2, \alpha_3) = (0.1, 0.7, 0.2)$

Comparative analysis: In this simulation, we compare the performance metrics in “w/o rating” and “with rating” marketplace settings to demonstrate the utility of privacy rating and its components. Table 4.4 gives the values of the parameters, defined in the section 4.5.1.1, used in this simulation unless stated otherwise. To observe the utility of price preference; we considered the provider's WTA amount as the percentage of the LCD_{max} for LCD requests and HCD_{max} for HCD requests.

Fig. 4.17 depicts the instances of a single iteration of simulation for a pragmatist provider. Fig. 4.17a illustrates the buyer's sample space where each circle represents the buyer's LCD and HCD purchase requests. We plot the buyers' overall privacy rating on the x-axis and the criticality of their purchase requests on the y-axis. The size of the circle represents the buyer's WTP amount. Next, we execute the algorithm 3 to obtain the accepted and leaked purchase requests. Fig. 4.17b depicts the selection space of the provider comprising all the buyers' purchase requests whose privacy rating and WTP satisfy the provider's



(a) (b) (c)

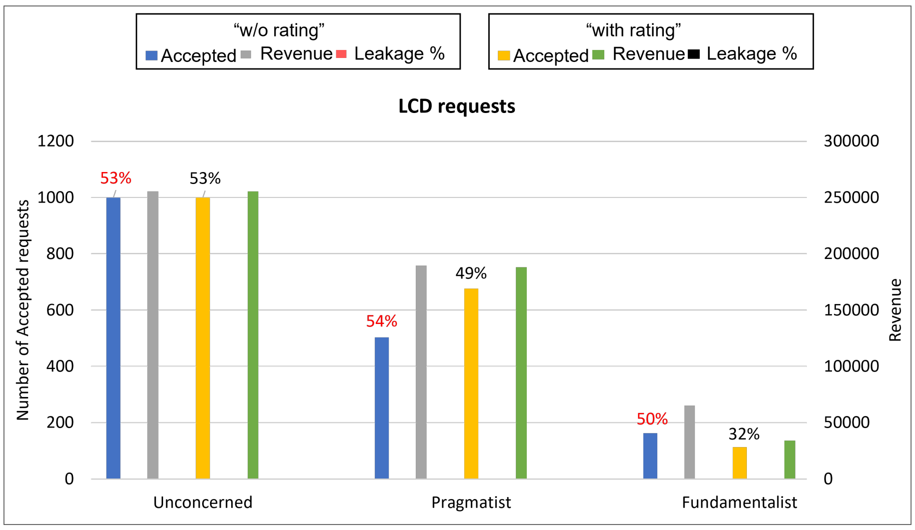
Figure 4.17: An instance of single iteration for pragmatist provider in "with rating" (a) Buyers sample space (b) Selection space showing accepted purchase requests (c) Affected requests due to data leakage

preference. Fig. 4.17c depicts the purchase requests affected by the data leak event.

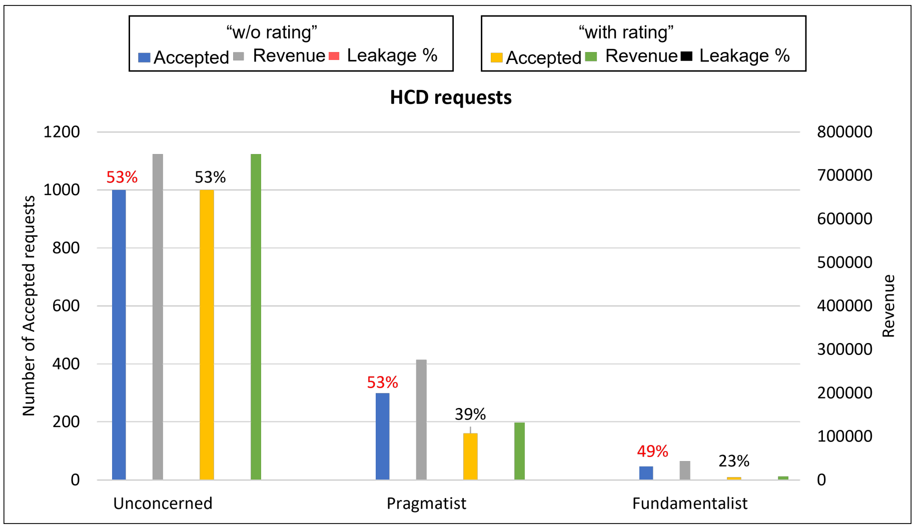
Fig. 4.18a and 4.18b compare the results for three providers in the "w/o rating" and "with rating" settings for (a) low critical data requests and (b) high critical data requests. The key findings are summarized below:

- Unconcerned provider: Since DP_U has the least privacy concern and lowest preferences among the three, all the purchase requests (LCD and HCD) received by him are accepted irrespective of the existence of a rating system. their revenue is more for HCD than LCD because HCD requests have higher WTP. Furthermore, the leakage percentage is estimated to be 53% in all the cases, which signifies that 53% of their disclosed data is leaked after the injection of the data leak event.

- Pragmatist provider: The number of accepted LCD requests increases from 503 in "w/o rating" to 677 in "with rating", while the revenue remains almost the same. Since we set the privacy preference of DP_P to 0.5 and relaxed their price preference from 50% to 10%, this expanded their selection space by including requests with low WTP. The revenue remains almost the same because requests of buyers with a privacy rating greater than 0.5 are selected who might have requested with low WTP. On the contrary, the number of selected HCD requests decreases from 299 in "w/o rating" to 160 in "with rating", and we observe a similar trend for revenue. Since



(a)



(b)

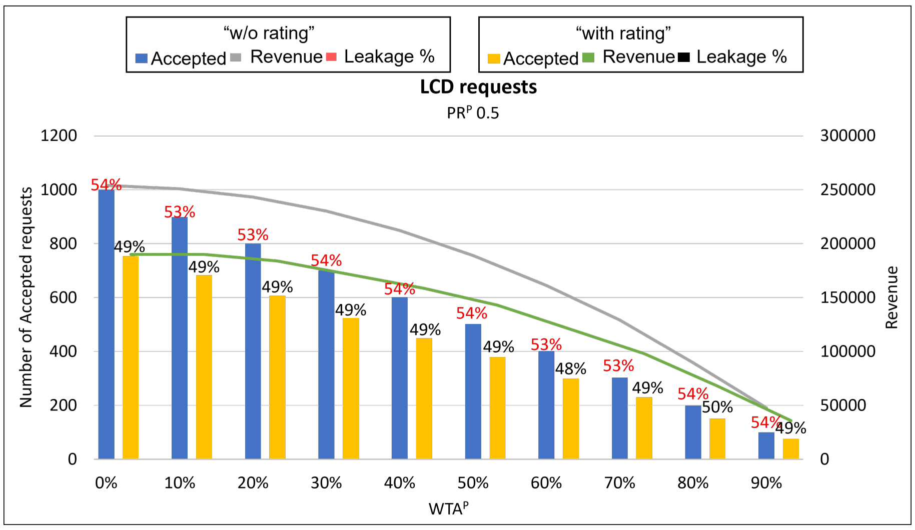
Figure 4.18: Comparative analysis of metrics in the "w/o rating" and "with rating" settings for (a) LCD purchase requests (b) HCD purchase requests

we set the privacy preference of DP_P to 0.7 and relaxed their price preference from 70% to 30%, many buyers with a lower privacy rating than 0.7 are eliminated from their selection space. The leakage percentage of DP_P decreases in "with rating" as compared to "w/o rating". A decrease in leakage percentage signifies that the amount of provider's data leaked is less in "with rating" than in "w/o rating". This is because DP_P is using privacy preference in "with rating" that eliminated all the buyers from their selection space whose likelihood of data leak is more. The decrease in leakage percentage is more for HCD (53% to 39%) than LCD (54% to 49%) because their privacy preference for HCD (0.7) is more stringent than for LCD (0.5).

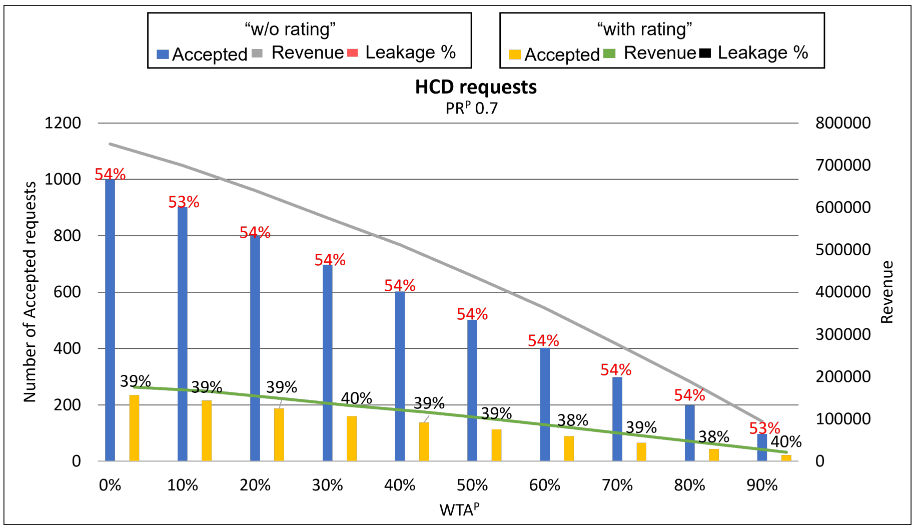
- Fundamentalist provider: The number of accepted requests in “w/o rating” decreases in “with rating” for both LCD requests (163 to 113) and HCD requests (46 to 9). We observed a similar trend for revenue as well. Compared to the DP_P, the number of accepted purchase requests and revenue is lower for DP_F. Since the fundamentalist provider exhibits the most conservative privacy attitude compared to the other two, their privacy preference considers fine-grained components of privacy rating (see section 4.5.1.1). With more filter parameters, their selection space shrinks, eliminating all the buyers who do not satisfy their preference. The number of accepted requests for LCD is more than for HCD because of lower privacy and price preferences choices of DP_F in the case of LCD. We observed leakage percentage decreasing for both LCD (50% to 32%) and HCD (49% to 23%). The decrement in leakage percentage is more for DP_F as compared to DP_P because DP_F's preference for leakage rating is 0.6 for LCD and 0.8 for HCD, which is more than the overall privacy rating preference of DP_P (i.e., 0.5 for LCD and 0.7 for HCD).

Next, we analyze the impact of increasing providers' price preferences while keeping their privacy preferences the same. The discussion on the evaluation is as follows:

- Pragmatist provider: Fig. 4.19 shows that with the increase in price preferences, the total number of accepted requests decreases for both HCD requests (1000 to 97 in “w/o rating” and 235 to 22 in “with rating”) and LCD requests (1000 to 100 in “w/o rating” and 753 to 76 in “with rating”). This is because “w/o rating” only considers price preference, and in “with rating”, the selection space is reduced due to the use of privacy and price preference. We observe a similar trend for revenue due to the same reason. However, the decline in leakage percentage is more for HCD requests (from 54% in “w/o rating” to 39% in “with rating”) as compared to LCD (from 54% in “w/o rating” to 49% in “with rating”). The possible reason is that



(a)

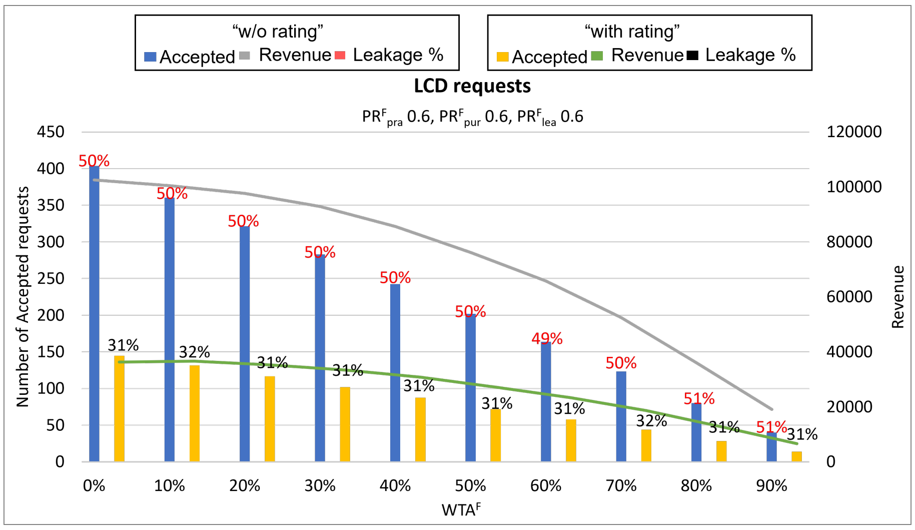


(b)

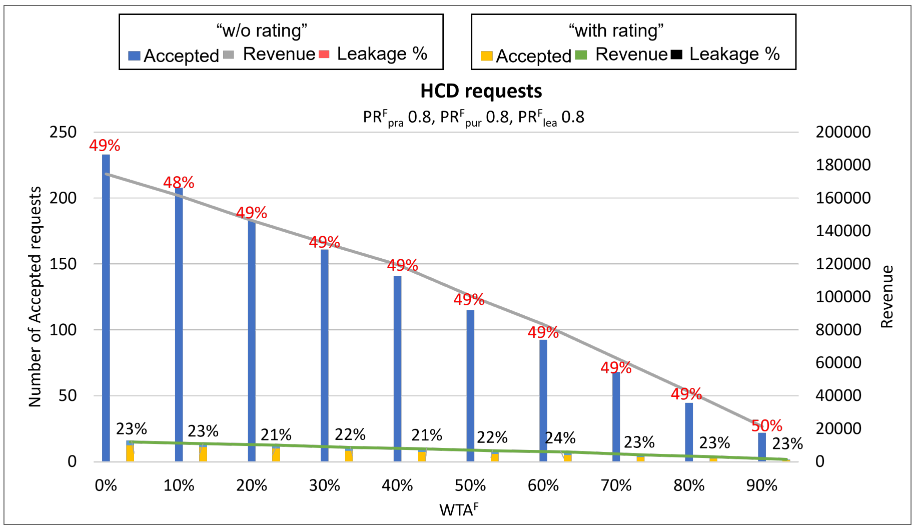
Figure 4.19: Effect of price preference for pragmatist in "w/o rating" and "with rating" for (a) LCD requests (b) HCD requests

since we use a more stringent privacy rating preference for HCD requests (0.7) than LCD requests (0.5) to define pragmatist provider preference, the buyers with a high likelihood of data leakage are possibly eliminated from the selection space; hence, a more significant decline is observed for HCD than LCD.

- Fundamentalist provider: Fig. 4.20 shows that with the increase in price preferences, the total number of accepted requests decreases for both HCD requests (233 to 22



(a)



(b)

Figure 4.20: Effect of price preference for fundamentalist in "w/o rating" and "with rating" for (a) LCD requests (b) HCD requests

in "w/o rating" and 16 to 2 in "with rating") and LCD requests (404 to 40 in "w/o rating" and 145 to 14 in "with rating"). Compared to DP_P, we observe that fewer purchase requests are accepted for DP_F in both settings because of a more stringent preference of DP_F than DP_P. We observe a similar trend for revenue as well as leakage rating for HCD and LCD requests in both settings. However, compared to DP_P, the decline in leakage percentage is more for DP_F, i.e., in the case of HCD

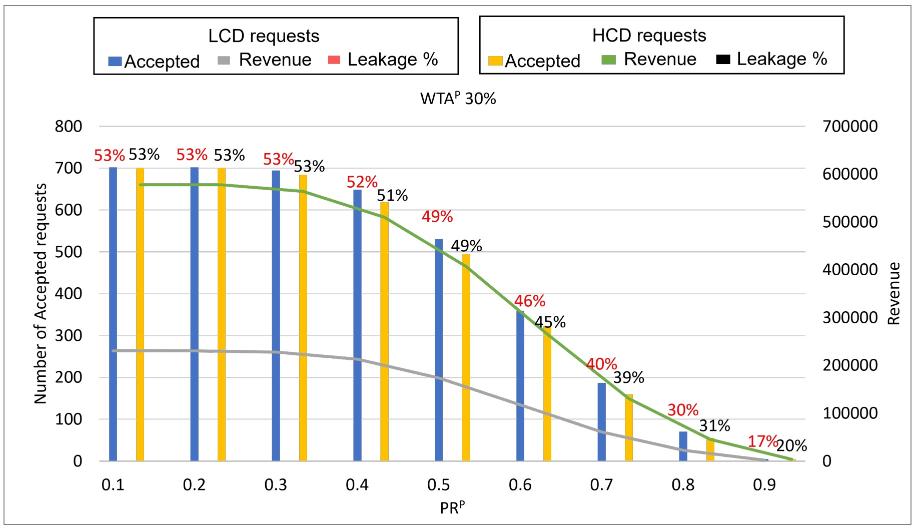


Figure 4.21: Utility of overall privacy rating

requests from 49% in "w/o rating" to 23% in "with rating" and in the case of LCD requests from 50% in "w/o rating" to 31% in "with rating". The possible reason is that DP_P considers the overall privacy rating in their preference while DP_F uses a fine-grained privacy component of privacy rating in their preference. Consequently, the selection space is reduced more for DP_F due to an increase in filter parameters. Moreover, buyers with lower leakage ratings, whose likelihood of leakage is more, are eliminated from the selection space, reducing the amount of leaked data.

An interesting observation from the above simulation is that with the increase in price preference, we observe no impact on the leakage percentage, which remains almost the same. This implies that price preference does not play any role in addressing the provider's privacy concern. We also observed although privacy preference reduces the provider's privacy concern, it impacts their overall revenue (utility). This observation aligns with the classical risk-utility trade-off providers must consider while trading their data in the marketplace. Therefore, we recommend the provider relax their price preference to increase their monetary benefits.

Next, we analyze the utility of privacy rating and its components by varying one parameter in the provider's preference while keeping other parameters the same.

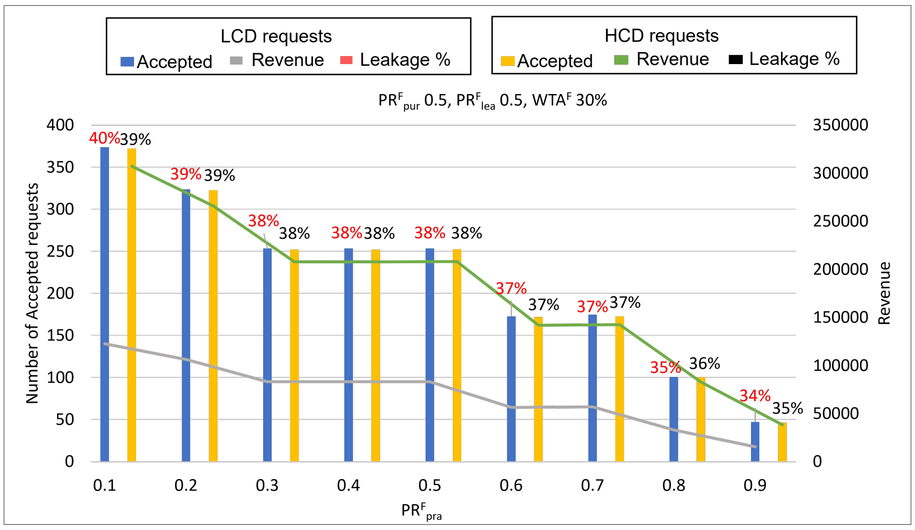


Figure 4.22: Utility of practice rating

Utility of overall privacy rating: Fig. 4.21 depicts the utility of overall privacy rating. We observe that for lower PR^{ij}, the number of accepted purchase requests remains almost the same for LCD (702) and HCD (701) requests. Accepted purchase requests gradually start decreasing with an increase in privacy rating. When PR^{ij} is greater than 0.3, the accepted requests for HCD slightly reduce as compared to LCD. Recall from section 4.3.2, the sensitivity of requested data is one of the factors used in determining the buyer's purchase rating. Therefore, HCD requests of buyers whose purchase rating is higher will only be accepted, resulting in fewer accepted purchase requests. The revenue for HCD requests is much higher than for LCD requests due to the higher WTP amount for high-sensitive data. We also observe the leakage percentage to be almost the same in both cases. Interestingly, DP_P can select HCD requests to increase their utility (revenue), and their risk (leakage percentage) remains the same as that of LCD. However, the number of accepted purchase requests will decrease slightly.

Utility of practice rating: Fig. 4.22 illustrates the utility of practice rating. We observe that with the increase in PR_{pra}, the number of accepted purchase requests decreases (374 to 47) for both LCD and HCD requests. Since we choose a uniform distribution to generate privacy elements of the practice profile, the computed practice rating of the buyers is possibly equally distributed in [0,1]. Hence, with the increase in PR_{pra}, purchase

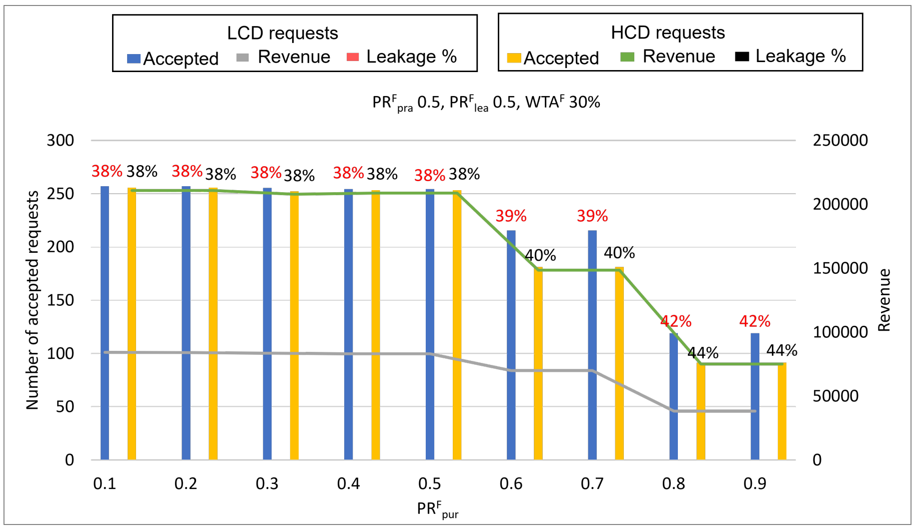


Figure 4.23: Utility of purchase rating

requests of buyers with high practice ratings are only present in the selection space, and buyers with low practice ratings are eliminated. Notably, the leakage percentage slightly decreases for LCD (40% to 34%) and HCD (39% to 35%). Recall from section 4.3.3, higher PR_{pra} implies that the buyer may be following good data security practices and hence, their likelihood of a data leak is low. This can be inferred as the data corresponding to their purchase request is not leaked due to the simulated data leak event.

Utility of purchase rating: Fig. 4.23 shows the utility of purchase rating. We observe that the number of accepted purchase requests remains the same (256) till $PR_{pur} = 0.5$ for both LCD and HCD requests. However, with a further increase in PR_{pur}, the number of accepted purchase requests for HCD decreases more than LCD. Recall from section 4.3.2, we considered the risk posed by the data purchase request in the computation of PR_{pur}. Since the privacy risk posed by the HCD requests is higher than LCD requests, the buyer's purchase rating is less for their HCD requests than their LCD requests. Hence, the selection space for HCD requests reduces more than for LCD requests. Interestingly, the leakage percentage increases with the increase in purchase rating. The reason is that since PR_{lea}=0.5, the buyers in the selection space have a higher likelihood of data leak; hence, the data corresponding to their purchase requests are leaked.

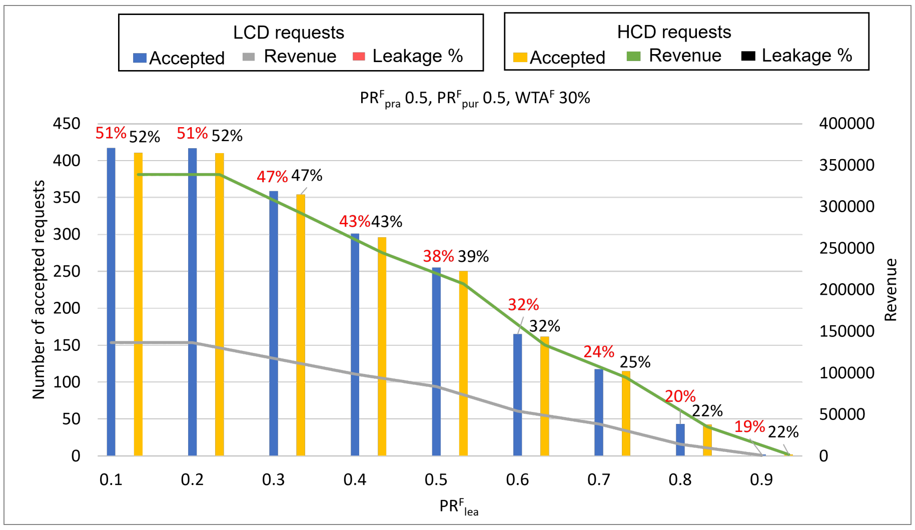


Figure 4.24: Utility of leakage rating

Utility of leakage rating: Fig. 4.24 depicts the utility of PR_{lea}. We observe that with the increase in PR_{lea}, the number of accepted purchase requests decreases for both LCD (417 to 2) and HCD (410 to 2) requests. This implies that fewer purchase requests of buyers satisfy the provider's preference with the increase in leakage rating. Interestingly, the leakage percentage decreases with the increase in leakage rating for both LCD (51% to 19%) and HCD (52% to 22%) requests. The reason is that a higher leakage rating implies buyers have a lower likelihood of data leak risk. Hence, data corresponding to their purchase requests is not leaked due to the simulated data leak event.

4.5.2 KYBChain performance evaluation

In this section, we provide a proof of concept implementation and overhead analysis of the KYBChain framework in terms of gas consumption, throughput and latency.

4.5.2.1 Proof-of-concept implementation

This section describes the proof-of-concept (POC) implementation of the KYBChain in a local private network based on Ethereum. We implemented various functions related to the

KYB-module and marketplace contained within smart contracts written in Solidity v0.8.9. However, Solidity v0.8.9 does not support floating point computation[1]. Therefore, the evaluation of privacy rating is performed off-chain while smart contracts are employed to record buyers' profiles in the blockchain. Off-chain evaluation of privacy rating can easily be validated using the proof stored in the ledger. We used the solidity web browser-based IDE "Remix" to develop these contracts. We used Python v3.7.4 scripts to simulate the interactions among contracts, web3 v5.30.0 library for communicating with the Ethereum node, and py-solc v4.2.0 library for compiling the smart contracts. Smart contract codes become immutable once deployed in the blockchain. Therefore, before implementing the POC, we cautiously performed unit and integration tests of the marketplace and KYB-module contracts on the Ganache Ethereum network.

We used Caliper v0.5.0 [234] as the benchmarking tool. For performance comparison, We consider a baseline marketplace system, MartChain (see Chapter 3), without a privacy rating system. The full stack of POC implementation, Caliper benchmark and smart contracts are available online[2].

The setup for POC is illustrated in Fig. 4.25. We defined a private Ethereum network that is built using Docker to simulate a real-world KYBChain. The network comprises two nodes and one miner, wherein each node runs geth v1.10.13 to manage the network. We bootstrapped the network by deploying all the smart contracts in the marketplace and KYB-module. To integrate the privacy rating system with the marketplace, the address of the deployed *PrivacyRatingSc* contract is updated in the *RegisterSc*. We modelled two classes: Admin and User, written in Python v3.7.4, that use web3.py v5.30.0 API for sending transactions to these networks. Admin class is used to create instances of auditor and regulator. User class is used to create instances of provider, seller and buyer. Each participant instance is assigned a unique Ethereum address. We initialized the setup by registering all the participants. The following steps describe the end-to-end data trade in the KYBChain implementation:

[1]https://docs.soliditylang.org/en/v0.8.9/

[2]https://github.com/pooja239/KYBChain

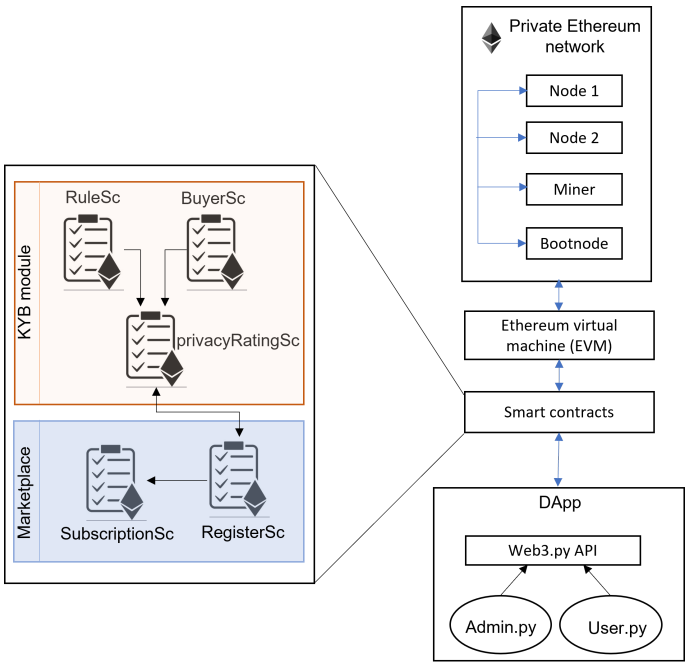


Figure 4.25: POC setup based on a private Ethereum network

1. When a buyer and provider comes under the agreement, a new *SubscriptionSc* is spawned and deployed in the KYBChain. The contract is registered by issuing $Tx_{registerAgreement}$, a multi-sig transaction, issued by both buyer and provider to the *RegisterSc*.

2. The provider issues Tx_{addS} to *SubscriptionSc* to add and initiate a subscription. After verifying the subscription details, a buyer issues Tx_{startS}. The buyer's escrow account holds an amount equivalent to the agreed-upon price. This is followed by marking the subscription status as ACTIVE.

3. The provider sends the requested data directly to the buyer. On receiving the data, the buyer submits $Tx_{confirmDelivery}$ to *RegisterSc*, which initiates the payment settlement process. The payment is automatically transferred from the buyer's escrow account to the provider's account. Accordingly, the subscription is marked as FINISH in *SubcriptionSc*. Moreover, *RegisterSc* also triggers a $Tx_{updatePurchase}$ to the *PrivacyRatingSc* that updates the buyer's purchase profile with the subscription details.

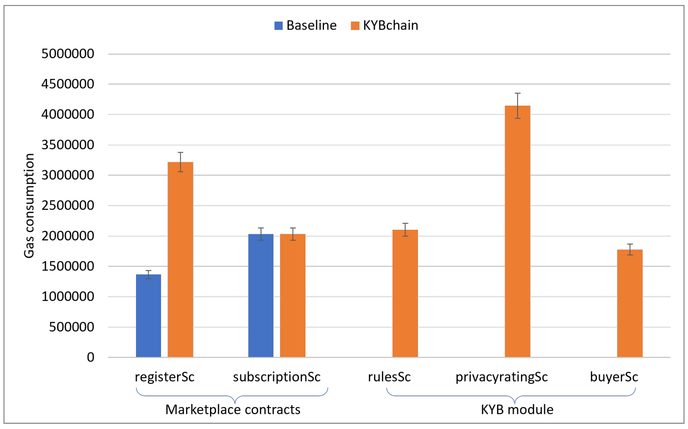


Figure 4.26: Deployment gas consumption

4. Steps 1-5 are repeated for all the new provider and buyer pairs. For every completion of a new subscription, the purchase profile of the associated buyer is updated.

5. Any user (provider, buyer, regulator, admin) can issue $Tx_{retrieveProfile}$ to retrieve the buyer's profile that can be used to calculate their privacy rating.

4.5.2.2 Performance evaluation

To illustrate the feasibility of our proposed architecture, we divide the performance evaluation into two parts: (1) gas consumption evaluation and (2) performance evaluation of smart contracts in terms of latency and throughput. We repeated all the testing scenarios for 10 iterations and calculated the average of the performance parameters.

Gas consumption: In our experiment, we record the deployment gas cost of deploying the marketplace and KYB-module smart contracts. Fig. 4.26 shows the deployment gas cost of *RegisterSc*, *SubscriptionSc*, *PrivacyratingSc*, *RuleSc* and *BuyerSc*. Compared to the baseline, the deployment cost of *RegisterSc* is more in KYBChain. This is because we added several administrator functionalities such as *addRegulator, registerAuditor, submitAudit* or *submitReport* and integration functionalities including *updatePurchase, updatePractice, updateLeakage* or *retrieveProfile* to integrate privacy rating system in the marketplace. Since there is no modification, the deployment cost of *SubscriptionSc* re-

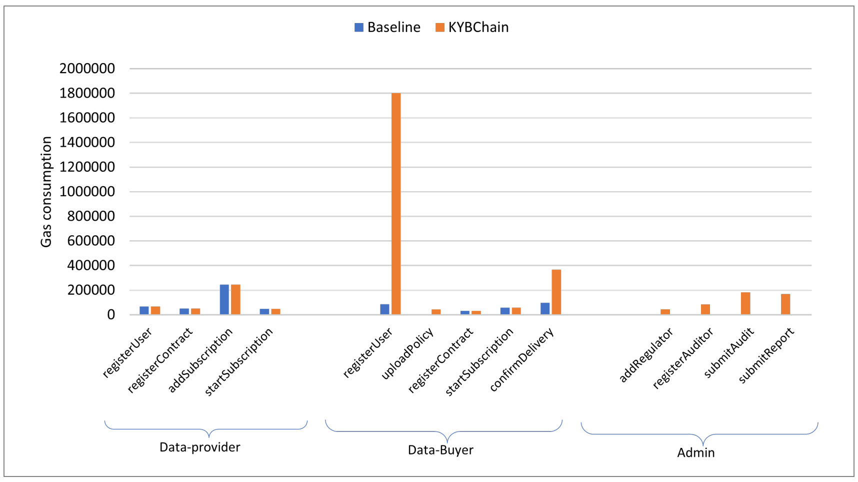


Figure 4.27: Tx execution gas consumption

mains the same in KYBChain as in the baseline. The deployment cost of *BuyerSc* and *RuleSc* is less than the *PrivacyRatingSc* in the KYB-module. This is because all the logic to retrieve buyer's profiles is implemented within *PrivacyRatingSc* while *BuyerSc* only manages the data storage in the ledger and *RuleSc* only manages the rules specifications.

We also evaluated the execution gas cost incurred by the provider, buyer, or admin by calling write functions in each smart contract during different stages of data trading. We implemented different functions in KYBChain with role-based permissions for execution (e.g. modifier to restrict access to certain functions) as shown in Fig. 4.11. Fig. 4.27 shows the granular gas consumption for the execution of different functions. It is observed that the *registerUser* in the *RegisterSc* for the buyer is the most expensive transaction in terms of gas consumption. This is because when a user registers himself as a buyer, an instance of *BuyerSc* is spawned from *PrivacyRatingSc*. Consequently, *registerUser* includes the deployment gas of *BuyerSc*. However, this transaction is infrequent since registration is just a one-time activity for any new buyer. As compared to the baseline, except $Tx_{confirmDelivery}$, all other transactions consume the same amount of gas. Recall from section 4.4.1.1, $Tx_{confirmDelivery}$ triggers the payment settlement, during which payment is transferred from the provider's account to the buyer's account. Post-settlement, *RegisterSc* issues $Tx_{updatePurchase}$ to *PrivacyRatingSc*. This transaction updates the purchase profile of the buyer with the subscription specifications, hence, consuming more gas than

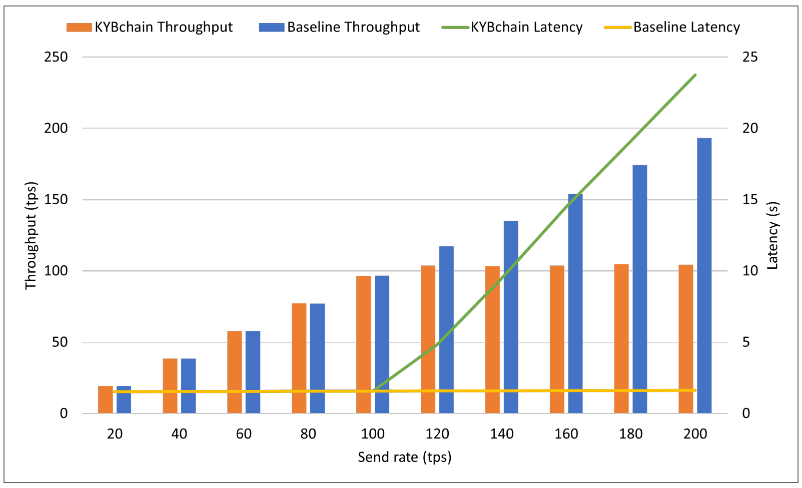


Figure 4.28: Evaluation of throughput and latency of $Tx_{confirmDelivery}$ in KYBChain and the baseline

the baseline.

Performance of Smart Contracts: As discussed previously, compared to the baseline, $Tx_{confirmDelivery}$ performs additional write operations to update the purchase profile of the buyer. Moreover, it is not only a state-changing transaction but is also frequent. Therefore, it is worth evaluating the computational overhead in terms of latency and throughput of $Tx_{confirmDelivery}$. Throughput is the rate at which transactions are committed to the ledger. Latency is the time taken from when a node sends the transaction in the network to the time it is committed to the ledger. We consider a base model of a solo miner node using Clique as the consensus mechanism from predefined network models in Caliper. Next, we present performance evaluations for $Tx_{retrieveProfile}$, which is a read-only transaction. Recall from section 4.4.1.1 that $Tx_{retrieveProfile}$ retrieves a buyer's practice, purchase and leakage profile. The purchase profile of a buyer retrieves their current possession and potential future acquisition specific to a provider. Recall from section 4.3.2, future acquisition retrieves the purchase profile of all other buyers to determine the provider's risk based on the data, buyer can potentially buy from them. Therefore, it is worth investigating the throughput and latency of $Tx_{retrieveProfile}$ with an increasing number of buyers in the KYBChain.

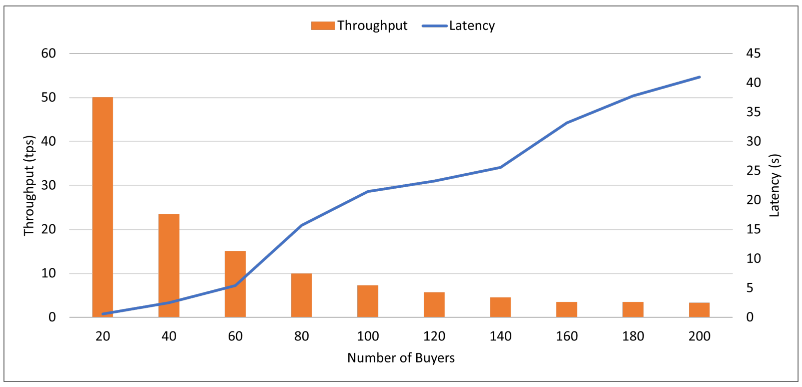


Figure 4.29: Throughput and latency for $Tx_{retrieveProfile}$

State-changing transactions: These transactions are broadcast to the network, processed by miners, and, if valid, are published on the blockchain. We performed the evaluations by varying the send rate of transactions from 20 to 200 transactions per second (tps) for a duration of 50s. Fig. 4.28 shows the evaluation results. It is observed that the throughput of $Tx_{confirmDelivery}$ increases linearly with send rate in both KYBChain and baseline. When the send rate approaches 120tps, throughput saturates at 104tps in KYBChain, while it keeps increasing in the baseline. Furthermore, latency remains low (1.5s) until 100tps in both cases and increases for KYBChain. The trend observed in KYBChain is due to the extra time taken to perform the additional write operations to update the purchase profile. This resulted in the congestion caused by the rapid growth in the number of transactions waiting in the execution and validation queues, which affects its commit latency.

Read-only transactions - Web3 API provides a call function for local invocation of a contract function that does not broadcast or publish anything on the blockchain. $Tx_{retrieveProfile}$ is a read-only operation for reading the buyer's profile from the ledger and returning it to calculate their privacy rating. Typically, the call transactions are synchronous, and the return value of the contract function is returned immediately. We initialize each buyer's purchase history by generating 10 purchases. We evaluate the throughput and the latency by increasing the number of buyers from 20 to 200 and issuing $Tx_{retrieveProfile}$ at a 50tps send rate for a duration of 50s. We repeated this test scenario for 50 iterations and calculated the average of the performance parameters.

Fig. 4.29 shows the evaluation results for $Tx_{retrieveProfile}$. When the number of buyers in the system increases from 20 to 60, the latency increases linearly to 5sec, and throughput decreases from 50tps to 15tps. This linear decline is due to the increase in the amount of time for executing each loop to retrieve future acquisitions. As buyers increases from 60 to 140, the latency gradually increases to 25sec, and throughput decreases further to 5tps. This is because as the number of buyers increases, the number of call transactions proliferates, and the node fails to retrieve the purchase profile for some requests; hence, transactions fail with the timeout error. However, when the number of buyers is close to 140, we observe that most transactions fail due to an "out of gas" error. Usually, read operations that return the state of the blockchain consume negligible gas compared to write operations. However, in our case, even though we declare these retrieve functions as *view* that does not change the state, they cost significant gas [235]. These functions in *purchaseRatingSc* call functions of different buyers' *BuyerSc* to retrieve their respective purchase profiles and execute *for* loops and *if* conditions. Therefore, when the number of buyers is 140, we double the default gas limit and increase the number of buyers further from 140 to 200. We observe a similar trend as observed earlier between 60 to 140. Further increasing the number of buyers resulted in an "out of gas" error again because of the high processing required for the same reason stated earlier.

4.6 Discussions and design implications

KYBChain presents a decentralized marketplace framework integrated with a privacy rating system for buyers. It enables providers to make a privacy-aware informed decision by estimating the risk of sharing their data with the buyer. KYBChain offers several benefits in overcoming the challenges mentioned in section 4.1. (1) Objectivity: Privacy rating is an objective metric based on factual and relevant information about the buyer. The provider can use the privacy rating to ascertain the implications of selling data to the buyer on their privacy. (2) Multifaceted and granular assessment: We define three different profiles, namely practice, purchase and leakage, to enable a multifaceted assessment of buyer's privacy rating to estimate the risk of non-compliance, data accumulation and data leak. We model these profiles using privacy elements that allow the adoption of different granularity levels based on the provider's preferences. (3) Automatic and dynamic: We proposed a stand-alone application, KYB-module, that comprises sets of smart contracts to automatically map buyers' activities in the marketplace with their profiles. (4) Auditability: KYBChain provides auditable and immutable records of buyers' profiles that regulators and policymakers can use and inspect, thus enhancing compliance. (5) Security: We carefully design access lists to enable role-based permissions for functions. Hence, restricting participants from unauthorized access and preventing any malicious entity from manipulating the rating. Moreover, KYBChain employs a hash-sharding technique to randomly select auditors for assessment. This prevents manipulation due to the collusion of buyers and auditors. Apart from these potential benefits of KYBChain, we also identify some limitations listed below that could be used as the basis for future research directions.

We use several rubrics or grading tables in our system to assess the risk magnitude of privacy elements while modelling the profiles. Such risk assessment varies considerably with the context, the metrics used as dependent variables, and the users' opinions. These models can be improved by adopting a qualitative method based on expert opinions and fuzzy techniques, as suggested in [219, 220] to calculate the level of non-compliance, accumulation and leakage risk associated with data sharing in the marketplace.

As highlighted in section 4.5, the throughput of the retrieval transaction ($Tx_{retrieveProfile}$) decreases as the latency increases with the increase in the number of buyers in the marketplace. Hence, with more buyers joining the marketplace, $Tx_{retrieveProfile}$ can be a KYBChain performance bottleneck. To overcome this issue, there is a need to optimize the implementation of purchase retrieval functions. One way of doing this is to utilize the underlying computing infrastructure for caching the frequently used data, such as the purchase profile of the buyers. The application monitors changes in the underlying ledger and updates the cached data with new state modifications [236, 237]. Instead of making frequent calls to the retrieval function, it can read the buyers' purchase profile from its application cache. Hence, improving the performance by serving future requests for profile retrieval faster.

Another limitation is the use of synthetic data in evaluating the efficacy of privacy ratings. Although the experimental results exhibit that the use of privacy rating tackles providers' privacy concerns, further generalization of the proposed approach can be explored using real-world data. Since privacy is a subject in which people often have different opinions, it would be worth conducting user acceptance surveys [230] to collect providers' data-sharing preferences.

4.7 Chapter Summary

In this chapter, we proposed a decentralized data marketplace integrated with a privacy rating system, KYBChain, which aims to provide secure, transparent, efficient and trusted maintenance and evaluation of buyer's privacy rating. The contribution of this chapter is two-fold. First, we proposed a privacy rating system to address the privacy concerns of providers related to non-compliance risk, data accumulation risk and data leakage risk. Corresponding to each risk, we maintain three profiles of the buyer, i.e. practice, purchase and leakage, based on their different characteristics. We identify privacy elements to model each profile. We use rubrics to score these privacy elements based on the associated magnitude of risk. We developed a methodology to map these risk scores to the buyer's rating. Second, we implemented the privacy rating system as a stand-alone blockchain-based application, the KYB-module, that keeps records of all buyers' profiles. KYB-module consists of smart contracts that automatically map buyers' activities with their associated profiles. Moreover, we also presented details of transactions and interactions of smart contracts during different stages of KYBChain. We performed a quantitative analysis to demonstrate the efficacy and utility of privacy rating based on three providers' privacy dispositions: unconcerned, fundamentalist and pragmatist. We also implemented a proof-of-concept on Ethereum and showed its effectiveness in terms of gas consumption. We performed an overhead analysis on Hyperledger Caliper to demonstrate KYBChain feasibility in terms of throughput and latency. Our analysis demonstrated the efficacy of privacy rating in aiding providers to make a privacy-aware decision about data sharing. The results showed that the use of privacy rating decreases leakage percentage and revenue, correlating with the classical risk-utility trade-off. Our evaluations of KYBChain revealed that the overheads introduced by our mechanism compared to a marketplace without a privacy rating system are insignificant relative to its privacy gains.

Chapter 5

TrailChain: Traceability of Data Ownership across Blockchain-Enabled Multiple Marketplaces

This chapter answers the research questions **RQ7**, **RQ8**, **RQ9**, and **IQ**. Despite the advantages of blockchain technology in decentralized data marketplaces as outlined in Chapter 5, a data marketplace may be prone to various threats. A provider may suffer privacy violation or monetary loss due to illegal or unauthorized reselling of their data. A malicious entity may trade unauthentic data, which can have severe repercussions on the buyer. A dishonest buyer can falsely claim the ownership of the purchased data causing misappropriation, forgery and identity theft for the provider. Traceability of data ownership is an effective approach to solve the above problems. Nevertheless, managing data ownership for traceability purposes is further complicated if a buyer buys data from one marketplace and, without the knowledge or consent of the provider, resells bought data to users registered in other marketplaces. Existing data ownership management approaches are not suited as current marketplace architectures are fragmented and lack mechanisms to track data ownership spanning multiple marketplaces. In this chapter, we propose TrailChain, a three-layered data ownership management framework based on a consortium blockchain network. TrailChain uses watermarking to generate a trusted trade trail for tracking the data ownership spanning multiple decentralized marketplaces. The novelty of TrailChain stems from (a) a mechanism for detecting any unauthorized data reselling within and across marketplaces; (b) a fair resell payment sharing scheme that ensures the resell revenue is shared with the data owners over authorized reselling; (c) a data ownership registration protocol that allows providers to register the ownership of original data and provides proof of data authenticity to the buyer by automatically verifying the trade lineage; (d) a prototype implementation using four private Ethereum networks and simulation to demonstrate TrailChain's feasibility by benchmarking performance metrics including execution gas costs, execution time, latency and throughput.

5.1 Introduction

The trading of IoT data promises many advantages for individuals, businesses and governments. However, in the wrong hands, the traded data might pose a privacy threat to the provider. In the previous chapter, we presented KYBchain, which employs a privacy rating system to evaluate buyers' privacy rating based on the risk associated with their different characteristics in the marketplace. Providers can use privacy ratings to make a privacy-aware data-sharing decision. Moreover, our findings show that a provider benefits from a privacy rating in minimizing their privacy risk due to data leaks (see Chapter 4). Unauthorized sharing of providers' data to third parties has been considered one of the major sources of their privacy concerns in the IoT data marketplace in [230]. Despite such benefits, a provider's revenue may decrease due to using privacy ratings, affecting their utility. A provider can manage this classical risk-utility trade-off [230] in two ways; either they restricts buyers from reselling their data, or they gives consent to buyers to resell their data legitimately to others in the marketplace. In return, the buyers can share their profit with the provider. Using the former method, a provider can control exposure to additional privacy risks by restricting the visibility of data but at the loss of utility. While in the latter case, even though the provider gets exposed to further risks, they can compensate for these risks by increasing their utility. However, some open challenges in the data reselling context, as discussed below, have either not yet been addressed or have only been partially addressed by the research community.

First, data ownership is ambiguous, as buyers may claim ownership of the purchased data. Unlike traditional physical commodities that are only owned by one specific entity, data ownership can include multiple parties as long as they have the ability to access the data and share it with others [117]. Hence, it is difficult to keep track of ownership once data is sold, which can lead to misappropriation, forgery and theft.

Second, data is an intangible asset [152, 238] and, therefore, can be resold in its original format or redistributed as a value-added service (VAS) without the knowledge of the provider. In cases when the provider may not want their data to be resold, illegal reselling or sharing may violate data privacy laws (e.g., GDPR). In other cases, when a provider is willing to share data for economic interests, unauthorized reselling leads to monetary losses. In light of these concerns, organisations such as the European Commission are contemplating the introduction of property rights over data, wherein a provider should receive a share of the economic value generated from their data or VAS [239–241]. With such laws and regulations, buyers will be required to share the generated revenue with the provider for purposes other than the stipulated use, such as the resale of the purchased data or VAS. Therefore, it is essential to facilitate a mechanism that allows providers to detect data resales and enable the distribution of the revenue generated by the resale of data among them.

For instance, Fig. 5.1 illustrates a data reselling and payment sharing scenario. A third-party location-based service provider ($Actor_4$) can provide campaign and promotional services to local businesses to increase sales. A simple solution to generate such customized promotional offers would be to buy users' cuisine preferences from food service applications ($Actor_2$), such as Zomato, and location data from geolocation collectors ($Actor_3$), such as [242] and run analytics on the same. The provider in this example is the individual ($Actor_1$) who generates and shares their original data with $Actor_2$ and $Actor_3$. $Actor_1$ should be able to detect when $Actor_2$ and $Actor_3$ resell their data to $Actor_4$ and receive a certain share p_{12} and p_{13} of the resell revenue p_{24} and p_{34} generated by $Actor_2$ and $Actor_3$,

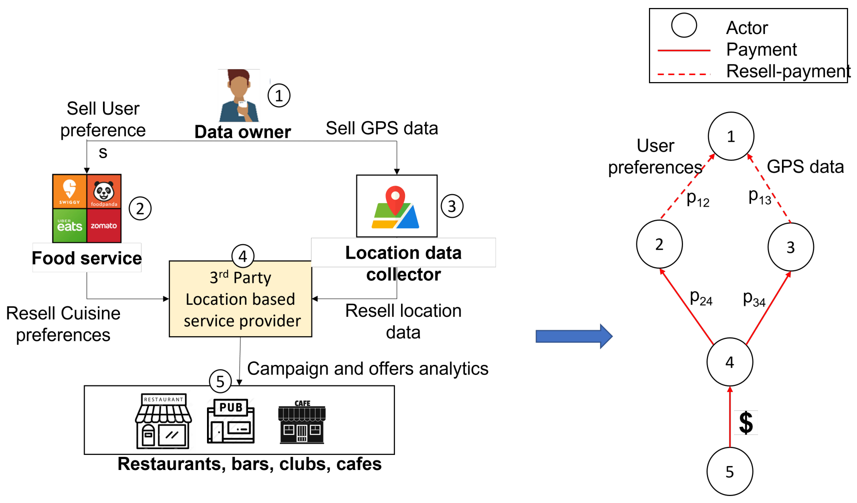


Figure 5.1: Data reselling and payment sharing scenario

respectively from $Actor_4$.

Third, it is challenging to achieve data integrity and ensure that the traded data is accurate, comprehensive, and sold with explicit consent. A malicious actor can perform various fraudulent activities, such as they can impersonate IoT devices to sell fake or bogus data or tamper with the data sent by legitimate devices, impacting the effectiveness and usability of such marketplaces. Thus, there is a need for a mechanism to ensure that the traded data is generated from a legitimate device and has not been altered through the entire lifecycle.

Fourth, it is likely that participants will register with multiple marketplaces for monetary gain or alternative offerings for data purchases. Users could buy data from one marketplace and resell the original data or value-added services to users registered in other marketplaces. Current marketplace architectures are fragmented and dispersed and lack mechanisms to track data ownership spanning multiple marketplace systems.

Digital watermarking technique [243, 244] has emerged as a promising way to address the challenges of illegal distribution or manipulation of digital data and for proving ownership. Recent work [245–248] has proposed methods for incorporating watermarks in time-series data such as sensor data or smartphone data. These schemes are designed to protect the digital copyright of the data by embedding the data owner's signature in the least significant bit (LSB) of data. In complex situations such as reselling where buyers also claim ownership of the purchased data, embedding several ownership signatures will require additional bits, hence, introducing significant distortions in the data and degrading the usability of IoT data. Besides, this approach does not preserve self-retrievable information of the sequence of data ownership.

In this regard, the joint use of watermarking and blockchain offers a promising way to prove and track ownership changes in distributed settings. Blockchain is essentially a distributed ledger that generates an immutable time-ordered history of transactions recorded in blocks. The watermarking technique embeds blockchain transaction ID and other public information in the data. This provides a secured and trusted linkage between the data and its modification or ownership history. This approach has been explored in some recent works [131, 137, 139] that fulfil a common interest of improving data ownership traceability. However, these solutions do not address issues such as ambiguous data ownership, undisclosed data reselling and dispersal of data ownership across multiple marketplaces.

TrailChain makes the following contributions:

- We propose a novel and fine-grained data ownership traceability mechanism called TrailChain, by leveraging the power of blockchain technology on top of existing watermarking techniques. Our framework provides an immutable and trusted trade trail for tracking the sequence of data ownership in an autonomous, efficient and transparent manner. TrailChain offers flexibility by identifying whether the resell/redistribution is legitimate/illegitimate or within or across marketplace systems.
- We devise a data ownership registration protocol that allows providers to register the ownership of original data and guarantees that the data has been generated by a genuine device. Our system employs digital notaries for facilitating data ownership traceability within and across marketplaces. Besides, it also provides proof of data authenticity to the buyer by automatically verifying the trade lineage. We also propose a fair resell payment sharing scheme that allows trusted, protected and

automated sharing of resell revenue among the data owners in the trade trail.

- Finally, we develop a prototype implementation of TrailChain on Ethereum and report detailed experimental results in terms of gas consumption and end-to-end execution time. We present a performance evaluation of the specific mechanisms introduced by our framework in terms of latency and throughput. We also perform a qualitative security analysis of TrailChain's resilience to various malicious activities. Simulations demonstrate that our method detects undisclosed reselling within the same marketplace and across different marketplaces. Besides, it identifies whether the reselling is authorized by the provider or not and fairly and automatically distributes the revenue among the data owners at marginal overhead. Finally, qualitative security analysis of the architecture highlights its effectiveness in providing immunity to several common attacks.

The rest of the chapter is structured as follows. Section 5.2 presents the overview of TrailChain. The detailed description of our proposed framework is presented in section 5.3. Section 5.4 presents proof of concept implementation, simulation results, and security analysis. Finally, section 5.5 provides discussions, followed by a chapter summary in section 5.6.

5.2 Overview of TrailChain

In this section, we outline the architecture of TrailChain from three aspects. First, the network model presents a multi-layer framework that integrates multiple decentralized marketplace networks and uses consortium blockchain to achieve secure, trustworthy, and transparent traceability of data ownership across these networks. Second, we explain TrailChain's architectural components. Third, we explain the high-level interactions of the entities within TrailChain.

Table 5.1 lists the notation used in this chapter.

Table 5.1: Table of notation used in this chapter

Notation	**Description**
MP_i	i^{th} Marketplace
ID_{MP_i}	i^{th} Marketplace identifier
DN_i	i^{th} Digital Notary
PU_{DN_i}	Public key of i^{th} digital notary
Sig_{DN_i}	Signature of i^{th} digital notary
DP	provider
Sig_{DP}	Signature of provider
PU_S	Public key of data seller
PU_B	Public key of buyer
PU_O	Public key of data owner
D	Traded Data
b_i	i^{th} batch of data sample in D
W_i	Watermark of batch b_i
SN_i	Serial number of batch b_i
d	IoT device generating data D
PU_d	Public key or identifier of IoT device d
CRP_d	Challenge-response pair of device d
C_d	Challenge of device d
R_d	Response of device d
P_d	One-way PUF functions of device d
N_a	Nonce
$T3(D)$	Trade trail table for data D
TID	Identifier of T3(D)
AID	Agreement identifier
SID	Subscription identifier
SS	Subscription state
PS	Payment state
P	Payment received by data seller on selling D
r_O	Resell payment ratio of data owner PU_O
p_O	Resell payment share of data owner PU_O

TrailChain contains the following actors: (i) providers who possess IoT devices that generate data and wish to monetize the original data, (ii) buyers that demand and purchase data, and (iii) data resellers that buy original data from providers or value-added services from other data resellers and resell them to other buyers to generate more revenue. In this chapter, we will refer to all the actors as data owners who have the ability to access and share data with others. Moreover, data owners can derive benefits from the data by selling original data or through value-added services. They also have the right to assign these access privileges to others. Therefore, the provider who generated the data D and all the buyers who have bought it are collectively referred to as the data owners of the data D.

5.2.1 Network Model

The network model of TrailChain is illustrated in Fig. 5.2. TrailChain consists of three layers: marketplaces, data ownership management and token management. Layer 1 consists of multiple separate decentralized marketplace networks $MP = \{MP_1, MP_2, ..., MP_N\}$

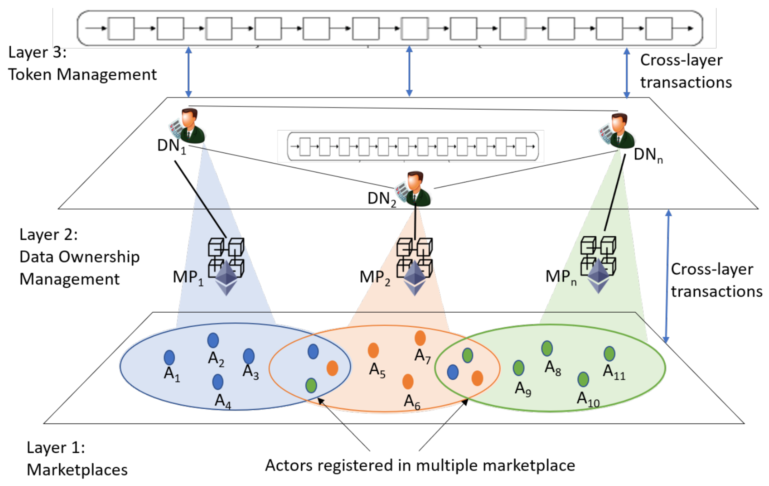


Figure 5.2: Network Model

that employ blockchain for managing and maintaining different trading-related functionalities. Each marketplace (MP_i) is associated with a set of data sellers and buyers with the following assumptions. Data sellers are interested in selling original data generated from their devices and are incentivized to participate in such a marketplace that can ensure tracking of further redistribution/reselling of their data. Buyers value genuine data and are incentivized to participate in such a marketplace that can ensure authentic data is sold fairly. We also assume that each participant is free to register himself in multiple marketplaces in TrailChain and can independently create offerings and trade data from one marketplace MP_i to another marketplace MP_j.

Layer 2 is based on a consortium blockchain network formed by a set of digital notaries $DN = \{DN_1, DN_2, ..., DN_N\}$ that enable secure tracking of ownership change when the data is traded within and across the underlying marketplaces. Each digital notary DN_i is associated with a marketplace network MP_i and facilitates verification of the trade legitimacy when data is traded in MP_i. Without a digital notary, data owners would be forced to use complex mechanisms to detect any unauthorized reselling of data as in [143]. Although the approach of letting each data owner perform ownership identification by himself is more secure, it requires them to behave honestly. Furthermore, the process of determining data ownership in [143] is similar to mining and suffers from high computational overhead. Hence, this approach is suitable only for a single reselling scenario in the same marketplace and not for multiple reselling scenarios in multiple marketplaces such as ours. Introducing a digital notary simplifies the requirements on the data owners for detecting illegitimate data trading. Additionally, a digital notary also notarizes the originality of the provider's data, provides proof of authenticity (POA) of the data bought by buyers and facilities resell payment sharing among the data owners. The role of the digital notary in TrailChain is carefully limited to the facilitation of aforementioned functions and automated by using smart contracts that implement these functions in an efficient, transparent and trusted manner.

The token management in Layer 3 manages and executes token transfer operations among data owners for facilitating payment settlement. In our model, we consider the trading of data across marketplaces, i.e., an actor can buy data in one marketplace and sell it to participants in another marketplace. Therefore, to enable resell payment sharing among the data owners, an asset exchange for the payment settlement among the participants registered in different marketplaces is required. To simplify this payment settlement, we assume that all participants register to a common token management network. We discuss the impact of relaxing the above simplifying assumption in section 5.5.

A digital notary also enables interoperability by facilitating cross-layer transactions wherein they reads and transfers the information recorded on the ledgers employed in each layer. To ensure trusted information exchange, we assume that each cross-layer transaction is accompanied with an attestation-based proof [129]. A proof provides the consensus view of the local network, and it requires a subset of members to attest their approvals of the information that is retrieved from their local ledger. The subset of members depends on the underlying consensus protocol employed in the respective layers. Before consuming the cross-layer information in the remote network, the proof is verified against the verification policy wherein each signature is validated, and each signer is authenticated using the configuration of other interoperable networks that is assumed to be priorly exchanged.

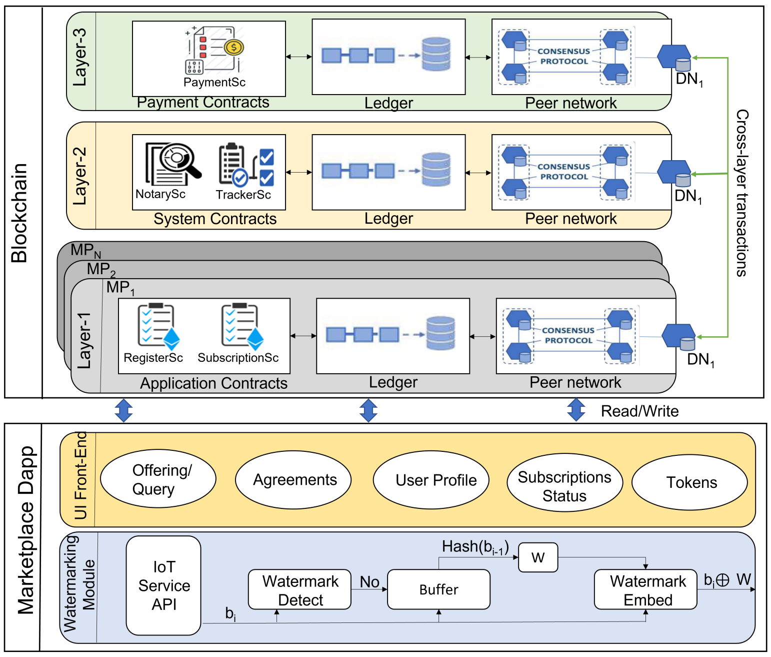


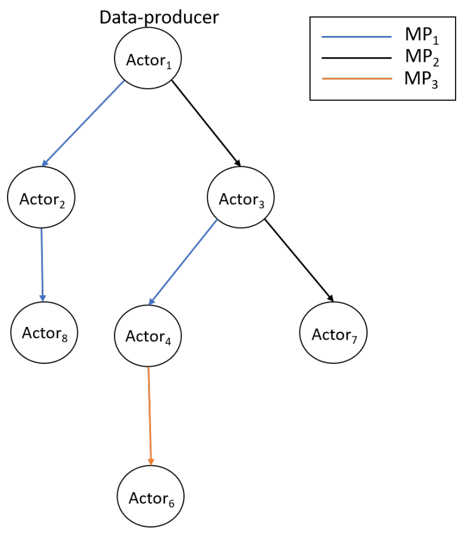
Figure 5.3: System Architecture of TrailChain illustrating multiple blockchain-based marketplaces integrated with data ownership management and token management via digital notaries

5.2.2 Architectural Components

The system architecture of TrailChain is depicted in Fig. 5.3. Each layer in the TrailChain employs a separate blockchain network and uses a set of smart contracts to implement specific functions. Layer 1 uses application contracts to provide marketplace functionalities such as management of user profiles and devices, agreements and subscriptions. Layer 2 employs system contracts responsible for tracking and managing ownership of data traded in the marketplaces. The payment contracts provide fair and secure resell payment settlement among the data owners in Layer 3. The framework also includes a marketplace decentralized application (Dapp) for participants to interact with the underlying marketplace, which also integrates a watermarking module to verify data integrity. The use of watermarking that allows the embedding of tracking details in the data itself and the inherited immutability and auditability of transaction records in blockchain provide secure, efficient and trusted data ownership traceability.

Layer 1 Application contracts: In this chapter, we adopt the application contracts design from Chapter 5 for demonstrating the integration of marketplace MP_i with TrailChain. However, TrailChain is agnostic of the underlying marketplace design and can similarly integrate other solutions such as distributed product catalogue [11], access control [54] for other marketplaces. Moreover, underlying marketplaces can be implemented in any instantiation of blockchain with adequate support for smart contract execution. The application contracts in MP_i provide an agreement framework using smart contracts to guarantee that the participants' behaviour automatically conforms to the terms of the agreements. The application logic is implemented by two main contracts: data subscription contract (*SubscriptionSc*) and register contract (*RegisterSc*). *SubscriptionSc* is deployed on the marketplace (MP_i) ledger for agreements made between each data seller and buyer pair and is only accessible to this pair. *RegisterSc* is unique to marketplace instantiation and provides core marketplace functionalities such as entity registration and agreement management. It uses various application binary interfaces (ABI) such as *registerUser*, *registerDevice*, *registerAgreement* for creating participants' profiles, registering their device and agreement in the marketplace network, respectively. Moreover, it also manages pertinent information such as the list of registered actors, description of devices whose data an actor is willing to trade, marketplace identifier (ID_{MP_i}) and associated digital notary (DN_i). To integrate the data ownership management framework with existing marketplace architectures, we extended the *RegisterSc* design by adding various interfaces. A provider uses *registerData* to initiate the data registration process for original data generated from their devices. The digital notary uses *ownershipRegistered* for adding data registration ID for a data registration request issued by a provider. Additionally, digital notary also uses *authencityRegistered* for recording Proof of Authencity (PoA) certificate, which is needed for data verification by the buyer. These newly added functions are designed with role-based permissions to restrict other entities from unauthorized access.

Layer 2 System contracts: System contracts are special sets of contracts that are written independent of marketplace application logic and deployed in Layer 2 for enabling data ownership traceability. Based on the transaction records in the Layer 2 blockchain, ownership of any data can be traced by every other digital notary. Therefore, the trade lineage



(a)

Owner ID	Resell-ratio	Buyer ID	Subscription ID	Marketplace ID
Actor$_1$	r_1	Actor$_3$	SID$_1$	MP$_2$
Actor$_3$	r_3	Actor$_4$	SID$_2$	MP$_1$
Actor$_1$	r_1	Actor$_2$	SID$_3$	MP$_1$
Actor$_4$	r_4	Actor$_6$	SID$_4$	MP$_3$
Actor$_2$	r_2	Actor$_8$	SID$_7$	MP$_1$
Actor$_3$	r_3	Actor$_7$	SID$_8$	MP$_2$

(b)

Figure 5.4: (a) Trade tree for data D (b)Corresponding T3(D): Trade trail table for data D

of any data can be proved and verified without any aid from a trusted third party. System contracts maintain and manage the immutable ownership-related information for each data (D) traded in the underlying marketplace networks in a trade trail table ($T3(D)$).

A $T3(D)$ of data D is defined as "a comprehensive history of data ownership". $T3(D)$ is identified by a unique trail identifier TID and records every trade transaction of data D in chronological order. Given a data D, its selling/reselling is modelled as a trade tree as depicted in Fig. 5.4a. The trade tree is represented as $T(V, E)$ where each vertex $Actor_i \in V$ is attributed to a specific actor who has the ownership of D. The root of the trade tree is the provider who owns the device from which the data was generated. Each vertex $Actor_i$ in the trade tree is uniquely identified by the actor's identifier PU_{A_i} implying that the same data D cannot be bought twice by the same actor. The edge set

E consists of directed edges E_{ij} representing a trade from $Actor_i$ to $Actor_j$. For each $Actor_{ij}$, a corresponding entry is formed in the $T3(D)$ as depicted in Fig. 5.4b.

When data (D) is traded from data seller (PU_S) to buyer (PU_B), the new ownership details are updated in $T3(D)$. Each entry in $T3(D)$ consists of data seller ID (PU_S), buyer ID (PU_B), marketplace ID (ID_{MP}) in which trading takes place, corresponding unique marketplace subscription ID (SID) and the data seller's resell payment ratio (r_S). A resell payment ratio (r_O) of a data owner (PU_O) is defined as the ratio of the payment received by the data owner ($payment_O$) from the immediate data reseller to the payment received by the data reseller on reselling the data ($payment_{RS}$), i.e., $r_O = payment_O/payment_{RS}$. A data owner or data seller defines their resell payment ratio (r_S) when a new subscription request is added to the agreement based on the negotiation with the buyer. The precise value of the ratio depends on various properties of the data, such as its demand, usefulness, quality, privacy risk [249] and storage cost.

System contracts consist of two smart contracts: *TrackerSc* and *NotarySc*. To facilitate secure smart contract updates, we separate the business logic and data storage-related functions into different smart contracts. The main smart contract *NotarySc* maintains the identity and configuration information for all marketplaces to identify the origin of the trade. It also contains functions for resell payment share settlement and trade lineage verification. It connects to the secondary contract *TrackerSc* to collect and store data. This is important when a bug is discovered and/or an update of a smart contract is necessary. A *selfdestruct* method is used to make the old smart contract unusable, and a new updated smart contract is deployed, and the accessibility of the old data remains intact through the new smart contract.

The *TrackerSc* contract uses the following data structure for recording the data ownership-related information in $T3(D)$.

```
struct Trail {
    bytes32 PU_S;
    bytes32 PU_B;
    bytes32 SID;
    uint r_S;
    bytes32 ID_MP; }
mapping(bytes32 => Trail[]) T3
```

NotarySc interacts with *TrackerSc* to retrieve the trade trail list of a data owner for data. A trade trail list is used to identify the trade flow from provider to data owner in the purchase order. A trade trail list of an *Actor* for data D is an ordered list of tuples given as $(TTL(Actor, D)) = (PU_O, ID_{MP}, r_O)$. As an example, the trade trail list of $Actor_6$ for data D in Fig. 5.4(a) is $TTL(Actor_6, D) = \{(PU_{A_1}, ID_{MP_2}, r_1)$, $(PU_{A_3}, ID_{MP_1}, r_3)$, $(PU_{A_4}, ID_{MP_3}, r_4)\}$. The trade trail list of a data owner is used to determine the legitimacy of the data source and identifies illegal/unauthorized reselling and double-buying, as explained in section 5.3.5. Additionally, the trade trail is utilized in the fair and secured distribution of resell payments among the data owners in the trade trail list. For a valid and legitimate trade trail, *NotarySc* uses $(TTL(Actor, D))$ to distribute payment P made by the buyer for the data D to the data seller and among the data owners in the trade trail list. It returns a payment share list (PSL) which is a list of tuple $\{PU_O, p_O\}$ where p_O is the resell payment share for data seller and each data owner in $TTL(Actor, D)$.

Layer 3 Payment contract: The proposed system defines a custom economic model based on a custom currency (Ethereum token), which can be used to interact with the underlying marketplace systems for payment settlement. Ethereum tokens are ERC20-compatible smart contracts that can act like a currency on the Ethereum platform. ERC20 [250] provides a standardised set of functions that must be implemented for tokens in Ethereum. Our model employs a *PaymentSC* smart contract for creating Ethereum tokens for transaction and payment purposes. We assume that all the participants (actors and digital notaries) create an account in Layer 3. The payment settlement is automated by employing escrow accounts in *SubscriptionSc*. To facilitate resell payment share for trades, the payment contract receives PSL from the digital notary along with attestation-based proof to verify and initiate the token transfer from the buyer's account to data owners as per the payment share list, as discussed previously.

Marketplace Dapp: The main function of the Dapp is to serve as the front-end to allow actors to interact with the main components of TrailChain. It uses the IoT service API to connect participants' IoT devices with the watermarking module used for embedding watermarks in the data generated from these devices. The Dapp also interacts with the application contracts to address queries and transaction requests from actors to record/retrieve data from the blockchain ledgers of different layers via digital notaries.

• **Watermarking module:** In TrailChain, digital watermarking is used to implement mechanisms for tracking the movement of data by enabling providers to embed hidden watermarks within data. Suppose data is found to be in suspicious possession or resold illegally. In that case, the embedded watermark in the data is used to identify the corresponding transaction trail recorded on the TrailChain to prove the ownership.

TrailChain is flexible to support any watermarking technique based on the data type. However, to prevent malicious buyers from stripping the raw data from watermarked data and illegally reselling it, our solution utilizes the existing backward watermark-chain data batch scheme [248]. In this scheme, data batches are chained with each other through a watermark, as explained below.

Before data transmission from provider to buyer, the IoT data $D = (b_1, b_2, b_3, ...b_N)$ is organized into constant-sized batches b_i. Each data batch consists of a fixed number of data samples, and if the last batch is shorter, zero-padding is used. For synchronization between the sender and the receiver, the batches are separated by a delimiter d that does not occur in the data readings. The watermark W_i of the data batch b_i is formed as:

$$W_i = HASH(TID|b_i|SN_i) \tag{5.1}$$

where $HASH()$ is a secure hash function applied on the concatenation of TID, data batch b_i and the batch serial number SN_i. Watermarking module in TrailChain uses trade trail identifier TID in Equation 5.1 that uniquely identifies immutable ownership history recorded in Layer 2, as explained earlier. Hence, providing a linkage between the traded data and its associated ownership details. Recall that TID is only known to authorized entities such as digital notaries. Since the attacker does not know the TID, any modifications to the data can be detected. The associated digital notary (DN_i) of

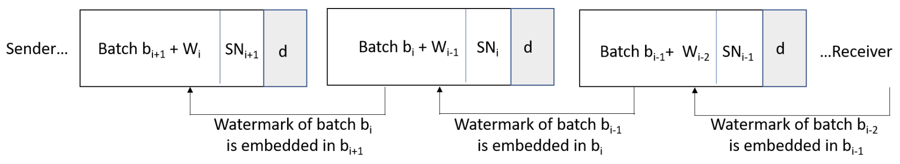

Sender...
Batch b_{i+1} + W_i
SN_{i+1}
d
Batch b_i + W_{i-1}
SN_i
d
Batch b_{i-1}+ W_{i-2}
SN_{i-1}
d
...Receiver
Watermark of batch b_i is embedded in b_{i+1}
Watermark of batch b_{i-1} is embedded in b_i
Watermark of batch b_{i-2} is embedded in b_{i-1}

Figure 5.5: Watermark-chained data

a marketplace (MP_i), where data D is traded, securely shares TID with the provider during the data ownership registration process, as explained in section 5.3.4. The use of SN_i prevents data reading insertion or deletion attacks. In our implementation, we use the $SHA3$ algorithm for hashing, which produces a fixed-length hash value (256 bits) for arbitrary-length input strings. Watermarking module uses two LSBs in each data sample in b_i, one bit for marking the presence of the watermark and another bit for embedding the watermark. Even in complex scenarios with multiple ownership details, this approach has a negligible distortion [247] and will not affect the usefulness of the data.

In the backward watermark-chain scheme, instead of sending the watermark W_i with the corresponding batch b_i of data readings, the watermark is embedded in the following batch (W_i of batch b_i is sent with batch b_{i+1}) as illustrated in Fig. 5.5. This way, the watermark is chained across all batches, making it difficult for an attacker to copy one or more data batches and replay them later [251]. This technique also ensures data integrity in an efficient, fast and lightweight manner at the application layer. Moreover, any change or tampering with the original data or an attempt to decouple data and watermark is easily detectable.

5.2.3 High-level interactions

The end-to-end interactions of the actors with the proposed framework are illustrated in Fig. 5.6. The process of selling data by *Actor*$_0$ (provider) to *Actor*$_1$ (buyer) is shown in Steps (1-7) while reselling of data by *Actor*$_1$ (data reseller) to *Actor*$_2$ (buyer) is shown in Steps (8-12). In Step 1, a *SubscriptionSc* is encoded and deployed in MP_1 when *Actor*$_0$ and *Actor*$_1$ come into an agreement. In Step 2, *Actor*$_0$ uses the data ownership

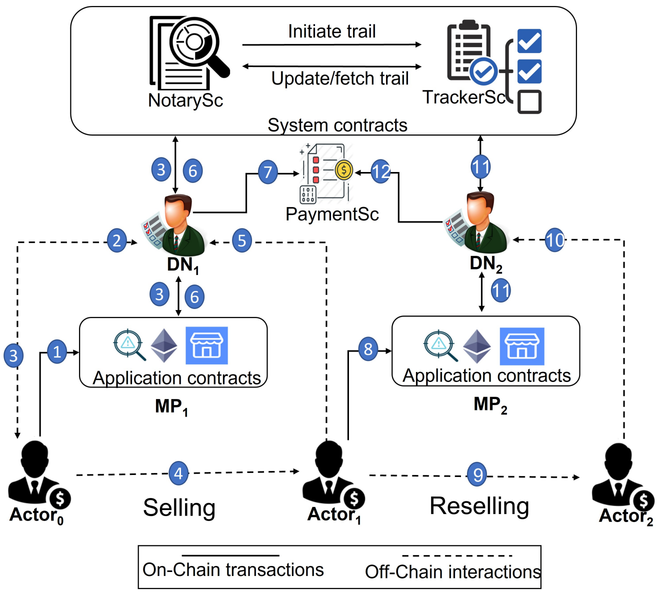


Figure 5.6: High-level interaction in selling and reselling scenario

registration protocol (see section 5.3.4) to register the generated data. In Step 3, DN_1 (digital notary of MP_1) creates a new trail in Layer 2, shares the encrypted TID and randomly-generated Nonce with the provider. DN_1 then records the hash of Nonce in the MP_1 ledger that TrailChain uses to validate the claim of ownership of original data by the provider. Post data ownership registration, TID is embedded in the data using the watermarking technique. Then, the watermarked data is transferred over an end-to-end encrypted channel (possibly using transport layer security (TLS)) in Step 4. The data exchange between two actors is assumed to be handled by the protocol chosen by the data seller. After receiving the data, $Actor_1$ initiates a data verification process with the DN_1 in Step 5. In Step 6, DN_1 reads the trade trail recorded in Layer 2, verifies the trade lineage, generates a proof of authenticity (POA) certificate, and records it in the MP_1 ledger. After receiving the POA, application contracts in MP_1 automatically initiate the payment settlement (Step 7), followed by payment share distribution in Step 8. In the case of reselling, a new *SubscriptionSc* is deployed in MP_2 in Step 8 when $Actor_1$ and $Actor_2$ come into a new agreement. $Actor_1$ transfers the data bought from $Actor_0$ to $Actor_2$ in Step 9. Steps (10-12) in reselling are similar to Steps (5-7) in selling, except instead of DN_1, all the intermediate transactions are handled by DN_2 (digital notary of MP_2).

Payment settlement uses a fair payment re-sharing scheme (see section 5.3.6) to calculate the resell payment amount for all the data owners in the trade linage and transfers the resell payment to the data owners using *PaymentSc*.

5.3 TrailChain Components

This section presents the details of the important components of TrailChain, which include the smart-contract-based approach for automated workflow execution, ownership traceability of traded data, data ownership registration and verification, and fair payment sharing. We will specifically cover six key components of TrailChain, as explained below.

5.3.1 Setup and Initialization

This component takes care of setting up marketplaces in Layer 1 and bootstrapping and initialising the data ownership management network in Layer 2. During the set-up phase, an existing or new marketplace network (suppose MP_i) is integrated with the TrailChain. As discussed earlier, the application contract, i.e., *RegisterSc* is extended with new functions such as *ownershipRegistered* and *authencityRegistered* that are required for digital notaries to interact with and fetch information from the associated marketplace ledger. To integrate an existing marketplace with the TrailChain, either a new version of application contracts integrated with new functions as stated above can be deployed, and all states from the old contract are migrated to the new one or existing application contracts can be upgraded by using update plugin of OpenZeppelin [227]. The problem of bootstrapping of data ownership management network (Layer 2) can be solved by incentivising participants for early adoption by having mining or service rewards using native tokens or by raising capital through an initial coin offering (ICO). This could be done by using the first phase of pure Proof-of-Work and then switching to Proof-of-Stake, like in PPCoin [252]. During the initialization phase, a digital notary (DN_i) registers himself to the marketplace (MP_i) and token management network to gain authorization to read/write in the respective ledgers. Next, they registers MP_i in Layer 2 to enable a data ownership traceability mechanism for every data trade happening in MP_i.

5.3.2 Registration of actors and their devices

We assume that TrailChain uses a common decentralized public key infrastructure (PKI) service across the three layers (i.e., data ownership management, marketplace, and token management) for enrolling and revoking public keys. All the entities (actors, digital notary and their devices) in the TrailChain are identified by their unique public keys across the three layers. In the rest of the chapter, we will use the actor's identifier and public key interchangeably. During the registration phase, an actor registers himself and creates a digital profile in the marketplace (Layer 1) and token management network (Layer 3) to record their public key. Actors use their private keys to sign their transactions. These signatures can be verified by the respective smart contracts, which have access to the public keys of the actors recorded in their digital profiles. In order to create data offerings in the marketplace from IoT devices, actors are responsible for registering their devices to the system using their unique parameters such as device serial number or MAC address [253]. In order to preserve privacy, this parameter is hashed, and only the hash value is recorded on the marketplace (MP_i) ledger as the unique identifier PU_d of the device d.

5.3.3 Agreement formation and execution

This component handles the workflow of subscription in an automated and non-repudiated manner. When a data seller and a buyer come to an agreement in a marketplace (MP_i), a new *SubscriptionSc* with the negotiated terms is spawned. It is encoded with details of the data seller and buyer, compiled and deployed in the marketplace ledger (MP_i) in Layer 1. The *SubscriptionSc* address and associated ABIs are registered to the *RegisterSc* using *registerAgreement*. *SubscriptionSc* maintains the subscription table that contains the following information for each subscription: subscription identifier (SID), data seller identifier (PU_S), buyer identifier (PU_B), device identifier (PU_d), data type, validation value, temporal context, total cost, subscription state (SS), verification value and payment state (PS). To enable ownership traceability, we include two extra fields for each subscription: validation value and verification value. These fields contain the outcome of data ownership registration and data verification processes, respectively, as explained in the following subsections, and can only be updated by the associated DN_i using $TX_{updateS}$. The structure of $TX_{updateS}$ is given below:

$$TX_{updateS} = [SID|Field|Value|Sig_{DN_i}|PU_{DN_i}] \tag{5.2}$$

where SID is the subscription identifier in the *SubscriptionSc*, PU_{DN_i} and Sig_{DN_i} are the public key and signature of the digital notary, respectively. The field corresponds to the field "validation value" or "verification value" while the value is the corresponding value.

All the other functions in *SubscriptionSc* related to adding subscription details, starting a particular subscription, etc. are only accessible by the data seller and buyer between whom the agreement is formed. This is to restrict external entities from interfering with the execution flow of subscriptions. Based on predefined workflows modelled by the data seller and buyer, *SubscriptionSc* manages and handles the operational sequence of each subscription using a separate variable *subscription state* (SS). Additionally, *SubscriptionSc* also employs an escrow functionality that uses another variable *payment state* (PS) for each subscription to automate the payment settlement when the buyer confirms the delivery of the data. *SubscriptionSc* automatically executes the steps of the subscription process following predefined business rules as explained next. When a new subscription is added, the SS is set to INITIATE while the PS is set to AWAITING_PAYMENT. If the data seller is the producer of the data, they will execute the data ownership registration process, during which SS changes to VALIDATING. At the start of the subscription, the SS changes to ACTIVE, and the buyer makes the deposit in the escrow account, which changes the PS to AWAITING_DELIVERY. Data is transferred off-chain from the data seller to the buyer over a secured connection. At the end of the subscription period, the buyer executes the data verification process, and the SS is set to SETTLEMENT while the PS is set to AWAITING_SETTLEMENT. Finally, the payment is transferred to the data seller and to the data owners as per the payment share list (PSL), and the receipt is sent to the buyer. Post-settlement, SS and PS are set to FINISH and COMPLETE, respectively.

5.3.4 Registering provider's data

As discussed in section 5.1, a data seller can easily generate inauthentic data instead of collecting authentic data from their IoT devices. It is hence difficult for buyers to verify whether data comes from registered data sources or illegitimate sources. Therefore, in our proposed mechanism, all the original data are required to be registered by the provider in the TrailChain prior to trading them in the marketplace. To this end, the provider is required to provide evidence to authenticate that the data is generated by a registered IoT device (d). TrailChain uses Physical unclonable functions (PUF) as evidence to establish trust at the origin. PUF acts as a unique hardware fingerprint of each IoT device. A PUF is formally described as a one-way function P_d that maps a set of challenges (C_d) to a set of responses (R_d) based on the physical microstructure of a device. Due to the naturally occurring manufacturing variations inherent to the fabrication process of the integrated circuits present in IoT devices, it is nearly impossible to modify, clone or tamper with a PUF [254]. A PUF can be represented as:

$$R_d = P_d(C_d) \tag{5.3}$$

A single challenge and the corresponding response of a device (d) are called the challenge-response pair and represented as CRP_d. The initial CRP_d is encrypted $CRP'_d = (Enc(C_d, PU_d), Enc(R_d, PU_d)$ with the public key of the device (PU_d). During the device registration process in marketplace MP_i, the provider also records the encrypted challenge-response pair of the device in the ledger of MP_i. Before transferring data generated from d, the provider registers the ownership of the generated data in the data ownership management network (Layer 2) via DN_i. We assume that the actors may be selfish and untrustworthy and may try to gain unfair monetary benefits by unauthorized reselling of data or selling ingenuine data. We propose a data ownership registration protocol (DORP) to prevent such dishonest behaviours, as illustrated in Fig. 5.7. DORP consists of the

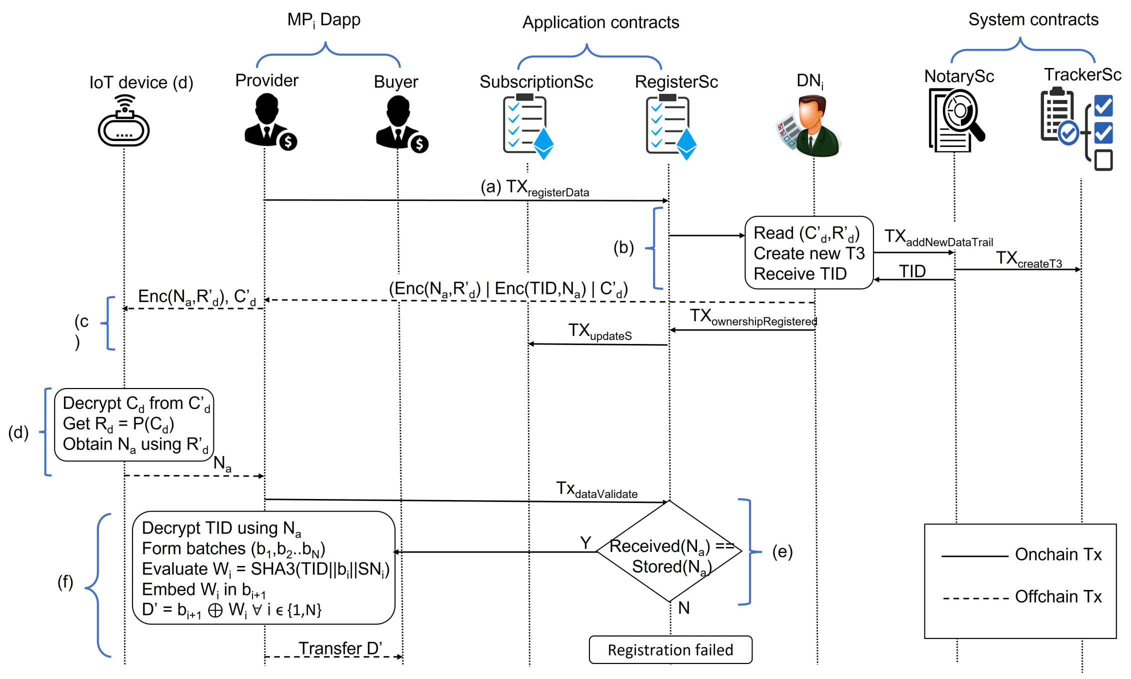


Figure 5.7: provider registering their original data in the TrailChain using DORP

following steps:

(a) Provider sends $TX_{registerData}$ (Equation 5.4) to *RegisterSc* for registration of data D generated from IoT device d for subscription SID. The transaction structure is given below:

$$TX_{registerData} = [AID|SID|Sig_{DP}] \tag{5.4}$$

where AID is the agreement ID to locate the *SubscriptionSc*, SID is the corresponding subscription identifier of the provider for trading generated data, and Sig_{DP} is the signature of the provider.

(b) For each new data registration request in MP_i, DN_i registers the ownership of new data in the TrailChain. DN_i locates the corresponding CRP'_d for d in the marketplace ledger (MP_i) using SID. If the CRP'_d is not found, the registration request is rejected. Otherwise, DN_i sends a $TX_{addNewDataTrail}$ (Equation 5.5) to *NotarySc* which in turn sends $TX_{createT3}$ to initiate $T3(D)$ associated with new data D. The *createT3* function returns TID that is used to identify the recorded trade trail in the data ownership management (Layer 2). The trade trail identifier TID is unique to each original data and can only be accessed by digital notaries via *NotarySc*. The transaction structure of $TX_{addNewDataTrail}$ is given below:

$$TX_{addNewDataTrail} = [ID_{MP_i}|SID|PU_{DP}|Sig_{DN_i}] \tag{5.5}$$

where ID_{MP_i} is the marketplace identifier and PU_{DP} is the public key of the provider, which can be used to identify the origin of the data.

(c) DN_i generates and encrypts a nonce value N_a with R'_d and TID with N_a and sends $\{Enc(N_a, R'_d), Enc(TID, N_a), C'_d\}$ to the provider's Dapp that forwards $\{Enc(N_a, R'_d), C'_d\}$ to IoT device d. Next, DN_i issues $Tx_{ownershipRegistered}$ to *RegisterSc*, which in turn sends $TX_{updateS}$ to update the validation value with $HASH(N_a)$ in the corresponding subscription identified by SID. The transaction structure of $Tx_{ownershipRegistered}$ is

$$TX_{ownershipRegistered} = [SID|HASH(N_a)|Sig_{DN_i}] \tag{5.6}$$

(d) Provider's IoT device d decrypts the challenge using its private key and obtains the corresponding response R_d for the challenge C_d using its PUF. Then using R'_d as the secret key, d obtains N_a and sends it to the provider's Dapp via IoT service API.

(e) The provider's Dapp then sends $TX_{dataValidate}$ as given in Equation 5.7 with the obtained nonce N_a to the *SubscriptionSc*. *SubscriptionSc* matches the received nonce N_a with the recorded nonce $Hash(N_a)$ in the marketplace (MP_i) ledger and marks the data validation status in the corresponding subscription SID as "validated" and returns success to the watermarking module in the provider's Dapp. Suppose the obtained nonce sent by the provider's Dapp fails to match the nonce recorded in the MP_i ledger. In that case, the registration fails and the validation status remains "invalidated" implying the origin of the data cannot be confirmed.

$$TX_{dataValidate} = [SID|N_a|Sig_{DP}] \tag{5.7}$$

(f) If $TX_{dataValidate}$ is successful, provider's Dapp uses the obtained N_a to decrypt TID. The watermarking module then generates a watermark using TID and embeds it in the data at the application layer, as explained earlier in section 5.2.2. Finally, the watermarked data is securely transferred to the buyer.

5.3.5 Data Verification Process for verifying trade lineage

The Data Verification Process (DVP) is illustrated in Fig. 5.8. After receiving the data batches $D = \{b_1, b_2 ... b_N\}$ from the data seller ($Actor_S$) in Step 1, the buyer ($Actor_B$) checks whether the data D is genuine, before initiating the payment settlement process. Since buyers are paying for the data, it is safe to assume that honest buyers are interested in checking whether purchased data items are genuine and have a legitimate trade lineage. Scenarios where the buyer intends to buy bogus or pirated data, are beyond the scope of this chapter. Besides ensuring data integrity, another benefit of using a backward watermark-chain technique is that buyers can select a subset of consecutive data batches to prove ownership. This approach significantly improves the network overhead compared

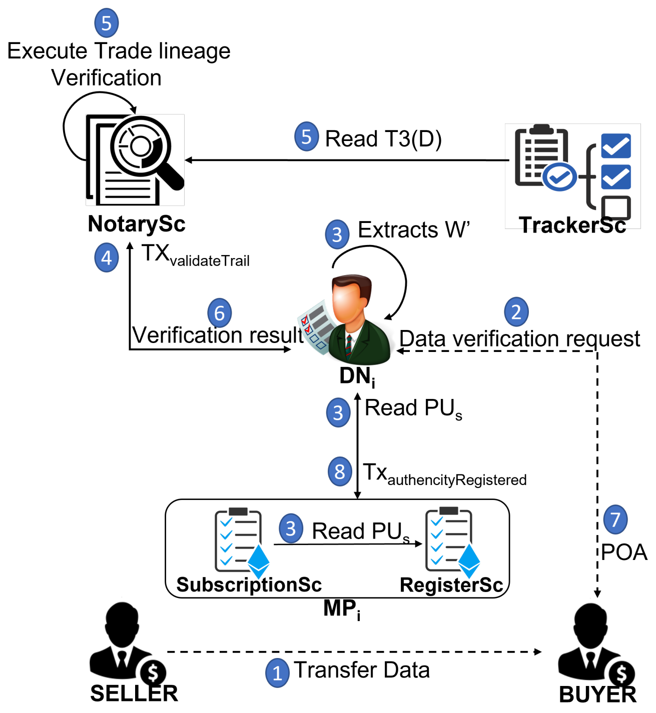


Figure 5.8: Data verification process initiated by buyer

to sending the entire data to DN_i for executing DVP. An honest buyer requests data verification by first choosing a subset of consecutive data batches $\{b_n, b_n + 1, ..b_m\}$ and sends them to the DN_i along with agreement ID AID and subscription ID SID in Step 2. In dishonest cases, a malicious buyer can send false or different data batches to the DN_i, which can impact the effectiveness of the data ownership traceability mechanism by resulting in wrong verification results. Such malicious activity can be addressed by employing a proof of delivery protocol [255], which can identify disputes in off-chain data transfer from the data seller to the buyer. Next, in Step 3, DN_i extracts the watermark W' from the received data batches using the watermarking detection and extraction algorithm given in [251]. Then, using received AID and SID, DN_i reads data seller's ID (PU_S) from the *SubscriptionSc* via *RegisterSc*. DN_i then sends the transaction $TX_{validateTrail}$ to *NotarySc* to verify trade lineage in Step 4. The transaction structures of $TX_{validateTrail}$ is given below:

$$TX_{validateTrail} = [TID|PU_S|PU_B|ID_{MP_i}|Sig_{DN_i}] \tag{5.8}$$

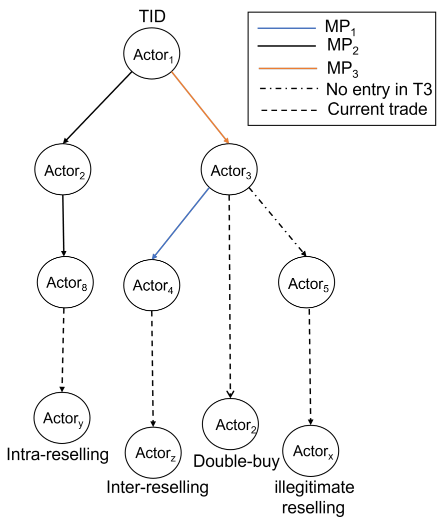
Then in Step 5, *NotarySc* executes the Trade Lineage Verification (TLV) algorithm as given in Algorithm 4. TLV takes seller's ID (PU_S), buyer's ID (PU_B), marketplace ID (ID_{MP_i}) where the trade happened and TID as an input. TLV uses $T3(D)$ corresponding to received TID to determine the trade legitimacy and returns the verification result based on the following four scenarios:

(1) Invalid data selling detection: Invalid data implies that no trade trail table $T3(D)$ exists in TrailChain identified by TID. No associated $T3(D)$ signifies that the provider has not registered their generated data D. An honest provider is expected to register their data in TrailChain using DORP for monetary gain or to prevent unauthorized data re-sharing. However, a malicious actor selling bogus/invalid data will intentionally not register their data in TrailChain. Such dishonest behaviour can be detected when the buyer executes the data verification process, and the malicious actor is referred to relevant authorities and could even be banned from the system.

(2) Illegitimate reselling detection: This can be explained using the reselling scenario from $Actor_5$ to $Actor_x$ as illustrated in Fig. 5.9a. The corresponding trade trail table ($T3(D)$)

Algorithm 4 Trade lineage verification

```
Input: $PU_S$, $PU_B$, $ID_{MP_i}$, $TID$
Output: result
1: Retrieve Trade trail table using $TID$
2: if (!$T3[TID]$.exist)
3:    result = "ERROR: Invalid data"
4: else if ($T3[TID].length$ == 0)
5:    if($PU_S$ != $dataProducer[TID]$)
6:       result = "ERROR: Illegitimate selling"
7:    else result = "No reselling"
8: else
9:    if($PU_B$ == $dataProducer[TID]$)
10:      doublebuy = true
11:   if ($PU_S$ == $dataProducer[TID]$)
12:      legitimate = true
13:   for $e$ in range(0, $T3[TID]$.length)
14:      if($T3[TID][e].PU_B$ == $PU_B$)
15:         doublebuy = true
16:      if($T3[TID][e].PU_B$ == $PU_S$)
17:         legitimate = true
18:   if(!legitimate && doublebuy)
19:      result = "ERROR: Illegitimate reselling and double-buy"
20:   else if(!legitimate) then result = "illegitimate"
21:   else if(doublebuy) then result = "double-buy"
22:      else result= "Success"
23: if(result == "Success")
24:   if($dataProducer[TID]$ == $PU_S$) then result = "No reselling"
25:   else retrieve $TTL(Actor_S, D)$
26:      market = $ID_{MP_i}$
27:      for $ele$ in range(0, $TTL(Actor_S, D)$.length)
28:         if(market == $TTL(Actor_S, D)[ele].ID_{MP}$)
29:            result = "Intra-reselling"
30:         else
31:            result = "Inter-reselling"
32:            break
33: return result
```



(a)

Owner ID	Resell-ratio	Buyer ID	Subscription ID	Marketplace ID
Actor1	r_1	Actor3	SID_1	MP_3
Actor3	r_3	Actor4	SID_2	MP_1
Actor1	r_1	Actor2	SID_3	MP_2
Actor2	r_2	Actor8	SID_7	MP_2

(b)

Figure 5.9: (a) Sample reselling scenario to explain illegitimate reselling, double-buy, intra-reselling and inter-reselling cases (b) Corresponding $T3(D)$

recorded in Layer 2 is depicted in Fig. 5.9b. A trade trail table identified by TID exists, implying the data is registered and verified by DN_i. However, no entry of the $Actor_5$, data reseller, in the $T3(D)$ is found. This can happen only when $Actor_5$ intentionally skips the data verification process for bought data. This situation may arise when either $Actor_5$ bought data illegally external to TrailChain, or they is unauthorised to sell the data. Once identified, dishonest data resellers are referred to relevant authorities and imposed heavy fines for such malicious activities.

(3) Double-buying: Double-buying happens when a buyer either unknowingly buys the same data multiple times from different data sellers, or a malicious data seller fraudulently sells the same data multiple times to the buyer. Double-buying can lead to buyer's resource wastage, hence, have a negative impact on the buyer's trading experience. Double-buying is detected by TrailChain when the buyer ($Actor_2$ as shown in Fig. 5.9(a)) is already listed

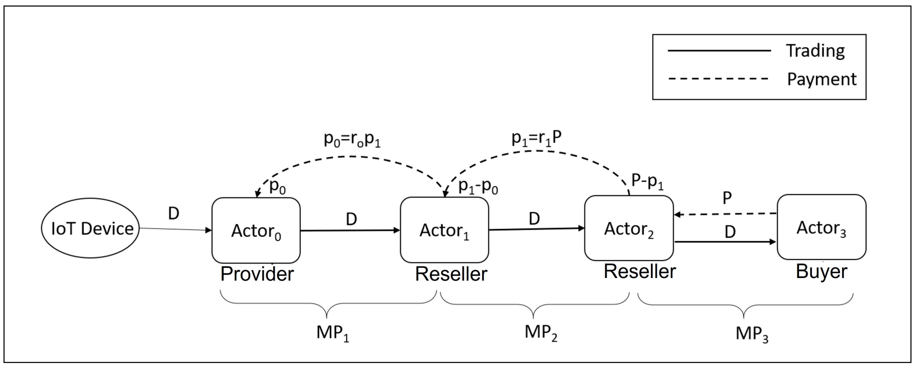
as a buyer in $T3(D)$ of the data (D) (refer to Fig. 5.9(b)). In such a case, the escrow account releases the payment back to the buyer.

(4) Authorized reselling detection: In the fourth scenario, a trade trail table associated with the key identifier is found and a valid trade trail list can be constructed for the data seller, which implies authorized reselling. We present the process for detecting authorized reselling under two cases: intra-reselling and inter-reselling. The reselling of data D among the actors residing within the same marketplace is referred to as intra-reselling, and reselling of data D across the marketplace is known as inter-reselling. In other words, the data ownership for D in intra-reselling remains within the same marketplace, while in inter-reselling, data ownership for D is dispersed in multiple marketplaces. For instance, as depicted in Fig. 5.9(a) and (b), a trade from $Actor_8$ to $Actor_y$ in marketplace MP_2 where $TTL(Actor_8, D) = \{(PU_{A_1}, ID_{MP_2}, r_1), (PU_{A_2}, ID_{MP_2}, r_2)\}$ is the case of intra-reselling. On the other hand, a trade from $Actor_4$ to $Actor_z$ in marketplace MP_1 where $TTL(Actor_4, D) = \{(PU_{A_1}, ID_{MP_3}, r_1), (PU_{A_3}, ID_{MP_1}, r_3)\}$ is an example of inter-reselling.

The trade is authorized and legitimate when the verification result of the TLV algorithm is either "No reselling", "intra-reselling", or "inter-reselling". In Step 7, for the authorized trading, only DN_i produces the proof of authenticity (POA) certificate and sends it with its signature to the buyer. Then in Step 8, DN_i executes $Tx_{authencityRegistered}$ in *RegisterSc* that updates verification value in the appropriate SID with issued POA. Consequently, data delivery to the buyer is automatically confirmed, and payment settlement is initiated.

5.3.6 Resell Payment Share Settlement

In accordance with the property rights over data wherein a data owner should receive a share of the economic value generated from their data or VAS as discussed in section 5.1, our resell payment share scheme fairly distributes the resell payment received by the immediate data reseller to each data owner in the trade trail. This can be explained by a sample scenario of reselling as illustrated in Fig. 5.10a. Suppose there are four actors



(a)

Owner ID	Resell-ratio	Buyer ID	Subscription ID	Marketplace ID
$Actor_0$	r_0	$Actor_1$	SID1	MP_1
$Actor_1$	r_1	$Actor_2$	SID2	MP_2
$Actor_2$	r_2	$Actor_3$	SID3	MP_3

(b)

Figure 5.10: (a) Sample payment-sharing scenario for reselling data D from $Actor_2$ to $Actor_3$ (b) corresponding T3(D) for sample payment-sharing scenario

$\{Actor_0, Actor_1, Actor_2, Actor_3\}$ where $Actor_0$ is the provider, $Actor_1$ and $Actor_2$ are the data resellers and $Actor_3$ is the buyer. Actors' association with the marketplaces is given as $MP_1\{Actor_0, Actor_1\}$, $MP_2\{Actor_1, Actor_2\}$ and $MP_3\{Actor_2, Actor_3\}$. The resell-payment share ratios for each actor recorded in TrailChain are r_0, r_1, r_2, r_3, respectively.

$Actor_3$ purchases the data D from $Actor_2$ by paying the price P. Using the $T3(D)$ recorded in Layer 2 as shown in Fig. 5.10b, trade trail list of $Actor_2$ for data D is retrieved which is given as $TTL(Actor_2, D) = \{(PU_{A_0}, ID_{MP_1}, r_0),\ (PU_{A_1}, ID_{MP_2}, r_1)\}$. Then using the resell payment share method, $Actor_2$ shares $p_1 = r_1 \times P$ with $Actor_1$ on receiving resell payment P from $Actor_3$. While $Actor_1$ on receiving resell payment p_1 from $Actor_0$, shares $p_0 = r_0 \times p_1$ with $Actor_0$.

Fig. 5.11 illustrates the resell payment share settlement process. *RegisterSc* issues an event to DN_i to initiate payment settlement. DN_i sends $TX_{evaluatePaymentShare}$ for the subscription (SID) to *NotarySc* that firstly verifies the trade lineage of $Actor_2$. Then based on the verification result, *NotarySc* sends $TX_{updateT3}$ to *TrackerSc* to update $T3(D)$ with the new data owner ($Actor_3$) information. Finally, trade trail list of data seller

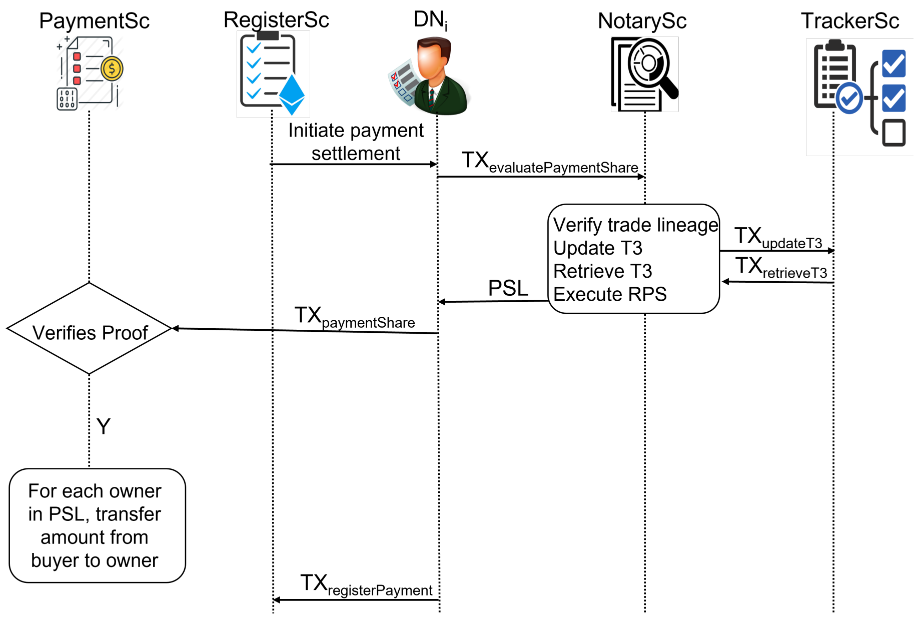


Figure 5.11: Resell payment share settlement scheme for trade in MP_i

($TTL(Actor_2, D)$) is retrieved from *TrackerSc* using $TX_{retrieveT3}$. It should be noted that new ownership detail is added in the $T3(D)$ during the final stage of subscription, i.e., the settlement phase. This is to ensure that $T3(D)$ is updated only for the authorized trading scenario while all the other discrepancies due to illegal reselling, off-chain data transfer, etc., are handled during the previous states of subscription. The transaction structure of $TX_{retrieveTrail}$ and $TX_{updateT3}$ are given as:

$$TX_{retrieveTrail} = [TID|PU_S] \tag{5.9}$$

$$TX_{updateT3} = [TID|PU_S|PU_B|r_S|ID_{MP_i}|SID] \tag{5.10}$$

Next, the retrieved $TTL(Actor_2, D)$ is used to execute resell payment share (RPS) Algorithm 5 that efficiently and fairly evaluates the resell payment share for each data owner in the trade trail list. For the trade ($Actor_S \rightarrow Actor_B$), the algorithm takes data seller $Actor_S$ ID (PU_S), their trade trail list $TTL(Actor_S, D)$ and payment P made by buyer $Actor_B$ as an input and returns a payment share list (PSL). Recall from Section 5.2.2, $TTL(Actor_S, D)$ is a list of tuple (PU_O, ID_{MP}, r_O) and PSL is a list of tuple $\{PU_O, p_O\}$.

Algorithm 5 Resell Payment Share Algorithm

```
Input: $PU_S$, $TTL(Actor_S, D)$, $P$
Output: $PSL$
1: $N = length(TTL(Actor_S, D))$
2: $P_{in} = P$
3: for $(j = N; j > 0; j - -)$
4:    $P_{out} = P_{in} * TTL(Actor_S, D)[j-1].r_O$
5:    $PSL[j].p_O = P_{in} - P_{out}$
6:    $P_{in} = P_{out}$
7:    if $(j == N)$
8:       $PSL[j].PU_O = PU_S$
9:    else
10:      $PSL[j].PU_O = TTL(Actor_S, D)[j].PU_O$
11: if $(N == 0)$
12:    $PSL[0] = (PU_S, Pin)$
13: else
14:    $PSL[0] = (TTL(Actor_S, D)[0].PU_O, P_{in})$
15: return $PSL$
```

The PSL for resell scenario in Fig. 5.10(a) is $\{(PU_{A_0}, p_0)$, (PU_{A_1}, p_1), $(PU_{A_2}, p_2)\}$. Finally, DN_i collects attestation-based proof from the subset of digital notaries in the data ownership management network (Layer 2) as evidence to prove the consensus view of the PSL. DN_i then sends a transaction $TX_{paymentShare}$ along with the proof to the token management network (Layer 3). The transactions structure of $TX_{paymentShare}$ is given as below

$$TX_{paymentShare} = [PSL|PU_B|Proof|Sig_{DN_i}] \tag{5.11}$$

where PSL is the payment share list, PU_B is the buyer identifier from whose account the payment is made, proof comprises of the signatures of a subset of digital notaries and Sig_{DN_i} is the signature of the DN_i associated with the marketplace (MP_i) where the trading happened.

To ensure a trusted and fair token transfer, $TX_{paymentShare}$ is confirmed when the proof is verified else it gets rejected. After confirmation, the payment P is transferred from $Actor_3$ to other actors in the PSL in which actors $\{Actor_0, Actor_1, Actor_2\}$ receive payments $\{p_0, p_1, p_2\}$ respectively. *PaymentSC* is also extended with implementations of the functions defined by the ERC20 standard, such as *balanceOf* and *transferFrom*, which return the token balance of an account, and enables token transfers between accounts, respectively. After the amount is transferred from the buyer account to the data owners in the PSL, DN_i notifies *RegisterSc* using $Tx_{registerPayment}$ the completion of the payment settlement and subscription. Finally, PS is set to COMPLETE, and SS is changed to FINISH for subscription ID SID in *SubscriptionSc*. The transactions structure of $TX_{registerPayment}$ is given as below:

$$TX_{registerPayment} = [SID|Sig_{DN_i}] \tag{5.12}$$

5.4 Evaluation And Results

In this section, we provide a proof of concept implementation, a quantitative performance evaluation and a qualitative security analysis of the TrailChain framework.

5.4.1 Proof-of-concept implementation

This section describes the proof-of-concept (POC) implementation of the TrailChain framework in a local private network based on Ethereum. We selected the private Ethereum blockchain as the implementation platform to show that our solution works for permissioned and permission-less blockchains. Our proposed architecture can be implemented in any blockchain instantiation that supports smart contract execution, for instance, Besu, Corda, Quorum and Hyperledger Fabric.

We implemented various functions related to ownership traceability and marketplace contained within smart contracts written in Solidity v0.5.13. However, Solidity v0.5.13 does not support floating point computation[1]. Therefore, we simulate resell payment share using integer values and define the resell ratio as percentages. We used the solidity web browser-based IDE “Remix” to develop these contracts. We used Python v3.7.4 scripts to simulate the interactions among contracts, web3 v5.2.0 library for communicating with the

[1]https://docs.soliditylang.org/en/v0.5.3/

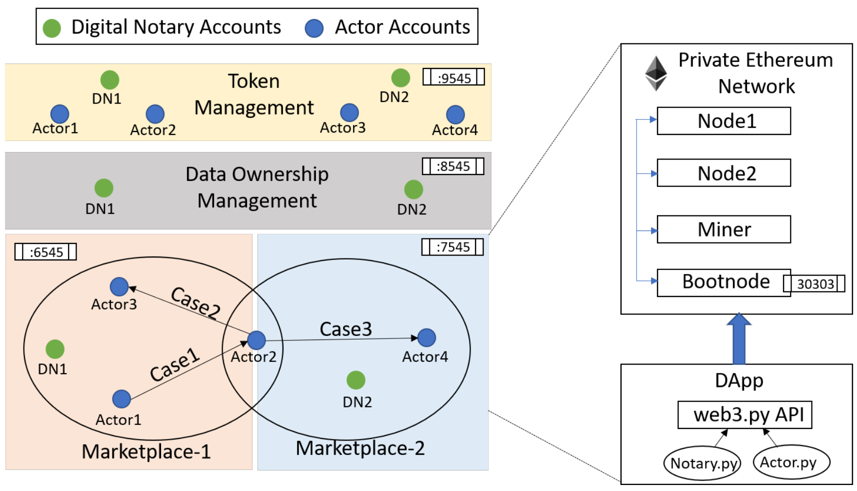


Figure 5.12: POC setup based on private Ethereum networks

Ethereum node, and py-solc v3.2.0 library for compiling the smart contracts. Smart contract codes become immutable once deployed in the blockchain. Therefore, we cautiously performed unit and integration tests of the application, system and payment contracts on the Ganache Ethereum network before implementing the POC.

We used Caliper v0.3.2 [234] as the benchmarking tool. All the performance tests are carried out on a 3.70GHz Intel(R) Xeon(R) 12, Linux Server (Ubuntu) with 62GB RAM Memory. For performance comparison, We consider a baseline marketplace system [249], without a data ownership traceability mechanism, which only provides agreement management functionality. The full stack of POC implementation, Caliper benchmark and smart contracts are available online[2].

The setup for POC is illustrated in Fig. 5.12. We defined four separate private Ethereum networks that are built using Docker to simulate a real-world marketplace. Each network consists of 2 nodes and one miner, wherein each node runs geth v1.10.13 to manage the network. To demonstrate the ownership traceability across marketplace networks, we use two separate blockchain networks, namely, Marketplace1 and Marketplace2, in which we deployed two sets of application contracts (*RegisterSc* and *SubscriptionSc*) to facilitate marketplace functionalities. The system (NotarySc and TrackerSc) and payment

[2]https://github.com/pooja239/TrailChain

(PaymentSc) contracts are migrated to two separate private Ethereum networks: the data ownership management layer and the token management layer. We modelled two classes: digitalNotary and Actor, written in Python v3.7.4 that use web3.py v5.2.0 API for sending transactions to these networks. We created two instances of digital notaries (DN_1 and DN_2) and six instances of actors ($Actor_1$, $Actor_2$, $Actor_3$, $Actor_4$, $Actor_5$ and $Actor_6$). $Actor_2$ is registered in both Marketplace1 and Marketplace2 to demonstrate the reselling use-case. These instances are linked with unique Ethereum addresses whose association with each network is depicted in Fig. 5.12. We initialized the setup by registering all the entities in the respective networks. All the actors' initial balance and resell ratio are set to 10000 tokens and 10%, respectively. Each subscription payment is 1000 tokens. We used GPS data consisting of 2 batches of 256 samples as an instantiation of streaming IoT data for demonstration purposes.

The different steps in the lifecycle of a data trade in TrailChain are shown in Fig. 5.13. Compared to the baseline [249] discussed in section 5.2.2, the trading in TrailChain involves three extra steps: data ownership registration process, data verification process and payment share evaluation and settlement. Recall that the term 'selling' refers to the trade of original data by a provider, while 'reselling' implies the trade of purchased data by a data owner. Therefore, the data ownership registration process (Step 4) is only part of selling and not reselling. Fig. 5.14 shows the snippet of the console output of the demonstration of our POC for selling between $Actor_1$ and $Actor_2$, intra-reselling between $Actor_2$ and $Actor_3$ and inter-reselling between $Actor_2$ and $Actor_4$. In addition, we also simulated two adversarial cases as discussed in section 5.3.5: (i) illegitimate reselling, where $Actor_5$ obtains the GPS data off-chain from $Actor_3$ and resells it illegitimately to $Actor_6$, and (ii) double-buying, where $Actor_3$ unintentionally purchases the same GPS dataset from $Actor_1$ that they had previously acquired from $Actor_2$. In both of these scenarios, the validation result is illegitimate resell and double-buying, respectively, and the payment settlement is not performed, as shown in Fig. 5.15. The complete console output of the POC is available online[3].

[3]https://github.com/pooja239/TrailChain/blob/main/POC/Output.txt

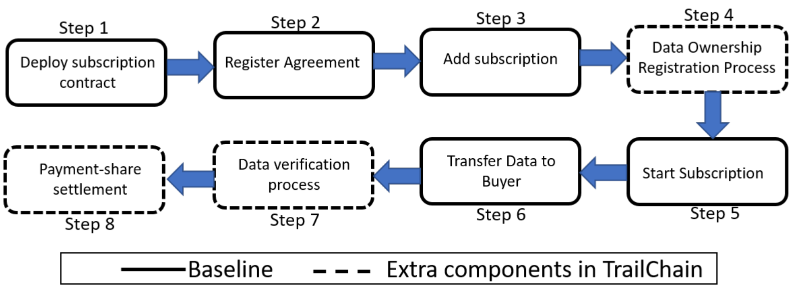
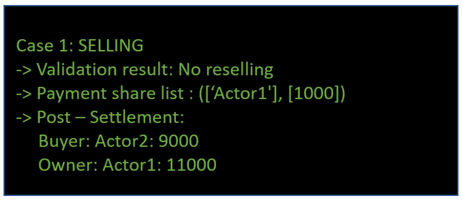
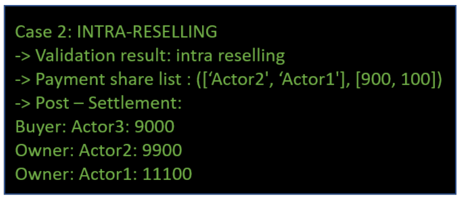


Figure 5.13: Lifecycle of a trade in TrailChain



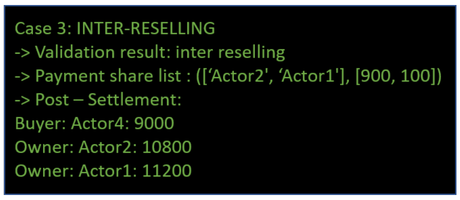
```
Case 1: SELLING
-> Validation result: No reselling
-> Payment share list : ([‘Actor1'], [1000])
-> Post – Settlement:
    Buyer: Actor2: 9000
    Owner: Actor1: 11000
```

(a) Case 1: Selling



```
Case 2: INTRA-RESELLING
-> Validation result: intra reselling
-> Payment share list : ([‘Actor2', ‘Actor1'], [900, 100])
-> Post – Settlement:
Buyer: Actor3: 9000
Owner: Actor2: 9900
Owner: Actor1: 11100
```

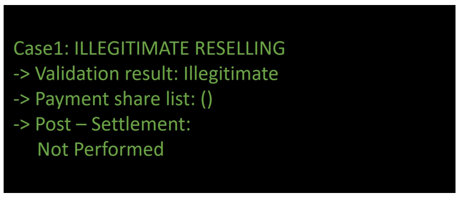
(b) Case 2: Intra-reselling



```
Case 3: INTER-RESELLING
-> Validation result: inter reselling
-> Payment share list : ([‘Actor2', ‘Actor1'], [900, 100])
-> Post – Settlement:
Buyer: Actor4: 9000
Owner: Actor2: 10800
Owner: Actor1: 11200
```

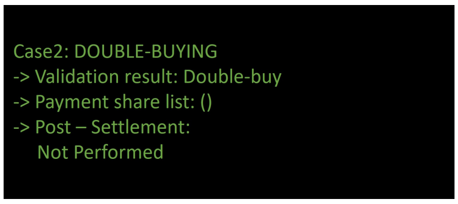
(c) Case 3: Inter-reselling

Figure 5.14: Snippets of the console output for the POC for three cases



```
Case1: ILLEGITIMATE RESELLING
-> Validation result: Illegitimate
-> Payment share list: ()
-> Post – Settlement:
    Not Performed
```

(a) Case 1: Illegitimate resell



```
Case2: DOUBLE-BUYING
-> Validation result: Double-buy
-> Payment share list: ()
-> Post – Settlement:
    Not Performed
```

(b) Case 2: Double-buying

Figure 5.15: Snippets of the console output for two adversarial cases

5.4.2 Performance Evaluation

To illustrate the feasibility of our proposed architecture, we divide the performance evaluation into three parts: (1) gas consumption evaluation, (2) end-to-end execution time analysis and (3) performance evaluation of system contracts. We used [249] as a baseline to compare the execution gas and execution time overhead in TrailChain. In addition, we

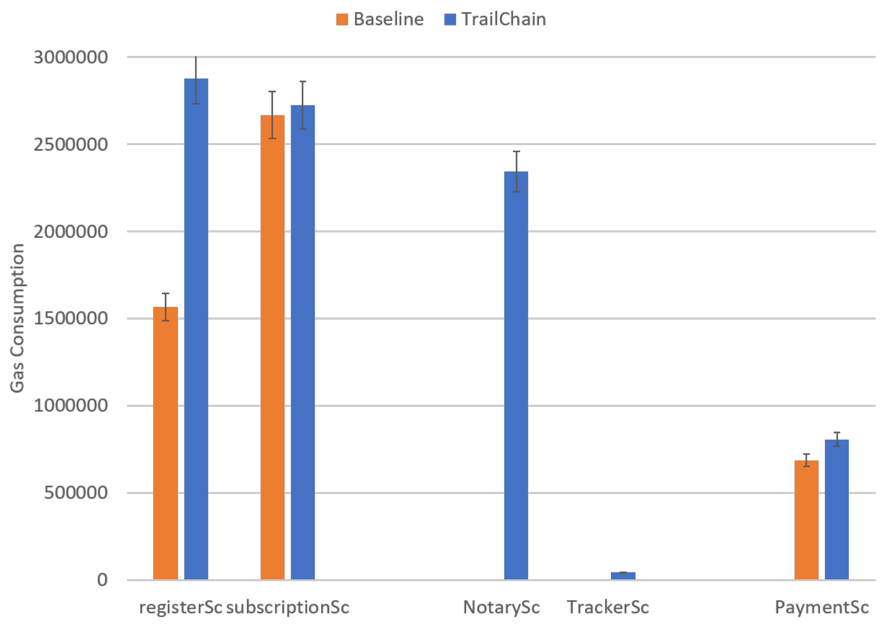


Figure 5.16: Deployment Cost

repeated all the testing scenarios in each evaluation for 10 iterations and calculated the average of the performance parameters.

(1) Gas consumption evaluation: In the Ethereum platform, any operation or transaction execution that changes the Blockchain or its state requires fees, and the involved parties need to pay the fees. These costs are calculated by using the amount of gas consumed that reflects the computational complexity of different transactions required in the trading process or the size of the smart contracts.

In our experiment, we record the deployment gas cost for deploying the relevant smart contracts in Layers 1, 2 and 3. We compare with the baseline introduced earlier, which does not include an ownership traceability mechanism. Fig. 5.16 shows the deployment gas cost of *RegisterSc*, *SubscriptionSc*, *NotarySc*, *TrackerSc* and *PaymentSc*. It is observed that the deployment cost of the application contracts is higher. Compared to the baseline, the deployment cost of *RegisterSc* is more because we added several registration functionalities, including *authercityRegistered*, *ownershipRegistered*, to enable ownership traceability in the existing marketplace application. The deployment cost of *SubscriptionSc* in TrailChain is

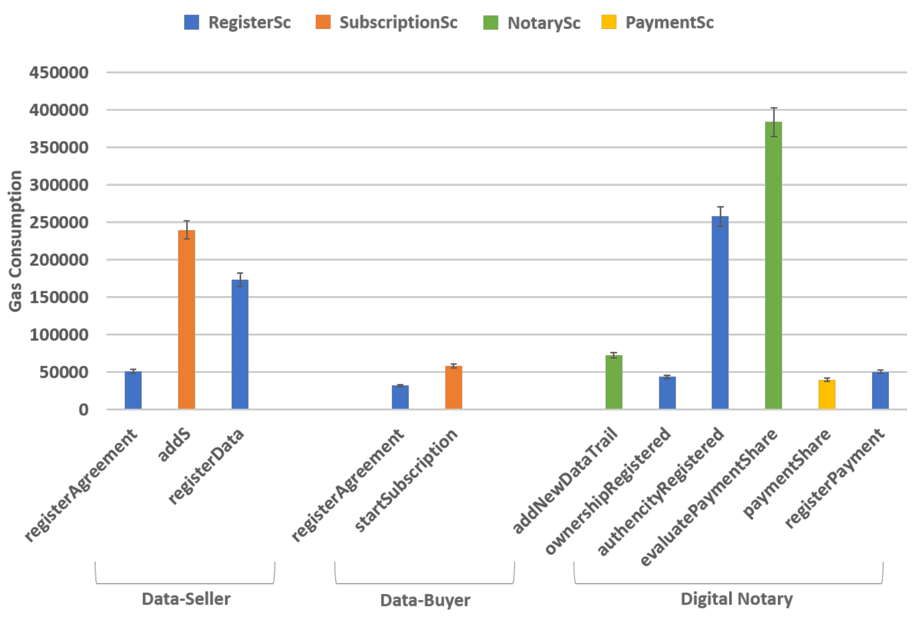


Figure 5.17: Tx Execution Cost

slightly more than the *SubscriptionSc* deployment cost in the baseline. This is because the subscription table contains two extra fields to record the data ownership registration and data verification results for the provider and buyer, respectively. The deployment cost of *TrackerSc* is less than the *NotarySc*. This is because all the logic to verify the trade trail and evaluation of payment share is performed within *NotarySc* while *TrackerSc* only manages the data storage in the ledger. Compared to the baseline, *PaymentSc* includes an extra function to distribute the payment among all the owners in the trail based on PSL, leading to a slight increase in the deployment cost.

We also evaluated the execution gas cost incurred by the data seller, buyer, and associated digital notary by calling write functions in each smart contract during different stages of data trading. We implemented different functions in TrailChain with role-based permissions for execution (e.g. using the require clause of solidity and variables such as msg.sender to check account authenticity). This ensures that the content can be retrieved and/or modified only by participants with specific roles. Fig. 5.17 shows the granular gas consumption for the execution of different functions. It is observed that the *evaluatePaymentShare* in the *NotarySc* is the most expensive transaction. Recall from section 5.3.6

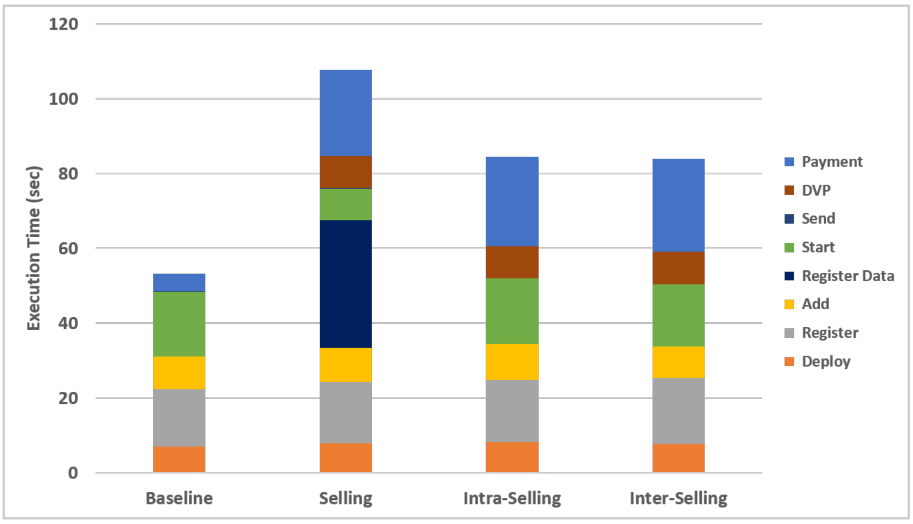


Figure 5.18: Execution time comparison

that *evaluatePaymentShare* provides the trusted payment share list (PSL) and performs three key operations, i.e., verification of seller's trade lineage, payment share evaluation and update current trade details in the $T3(D)$ in Layer 2. Note that *paymentShare* distributes the amount paid by the buyer to all the data owners in the trail and builds on *transferFrom* [250] and, therefore, does not require much gas.

(2) Execution Time Analysis: In this evaluation, we are interested in observing the increase in end-to-end execution time of trading in TrailChain due to the introduction of three extra steps as compared to trading in baseline: data ownership registration process (Step 4), data verification process (Step 7) and payment share settlement (Step 8) as illustrated in Fig. 5.13. End-to-end execution time is the time span of trade starting from *subscriptionSc* deployment till the payment settlement. Fig. 5.18 shows the end-to-end execution time for baseline, selling, intra-reselling and inter-selling cases. As expected, the execution time of trade in TrailChain is more compared to the baseline. Additionally, the execution time of selling is also more than the reselling cases due to the additional data-registration process in the former. However, the execution time of the different steps in inter-reselling and intra-reselling remain the same, implying TrailChain provides ownership traceability across marketplace networks with no additional time overhead compared to within the same marketplace network. It is worth noting that since there are no changes

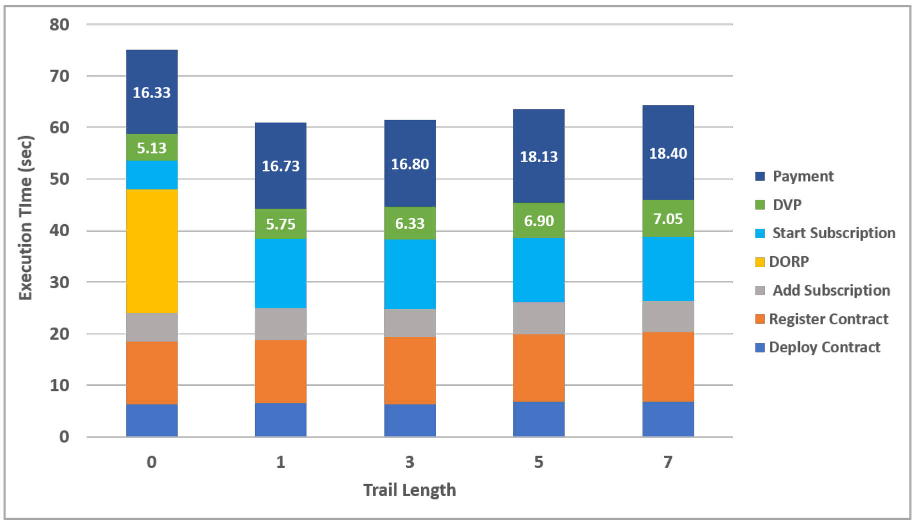


Figure 5.19: Impact of Trail length on execution time

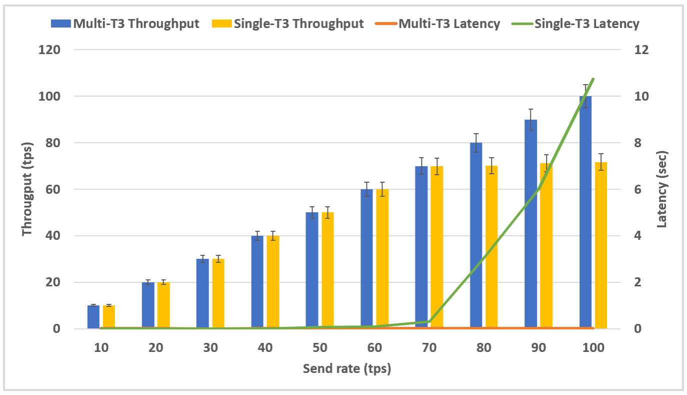
in the deployment of the subscription contract (Step 1), registration of the agreement (Step 2) and adding subscription (Step 3), execution time remains the same in all the scenarios. The time to execute the start subscription (Step 5) remains the same in the baseline and reselling cases, while it is lower in the selling case. This is because, at the end of the data ownership registration process (Step 4) in selling, the *RegisterSc* automatically activates the subscription for the provider. The execution time of DVP and payment share settlement (Steps 7 and 8) remains similar in selling and reselling cases. The execution time of payment in the baseline is less compared to TrailChain. This is because, during the settlement phase, the payment amount is transferred directly from the buyer to the data seller in the baseline. In contrast, payment share is evaluated and distributed among all the data owners in the trail in TrailChain.

Fig. 5.19 depicts the variation of the end-to-end execution time of a single trade depending on the trail length. Trail length is the number of data owners in the trail who have resold the data. A trail length of 0 corresponds to when the data is bought directly from the provider. It has the maximum execution time because of the data ownership registration process (Step 4). It is observed that the execution time of trade in the case of reselling increases with the increase in trail length from 1 to 7. However, the execution time remains almost the same in all the steps except DVP and payment share settlement (Steps 7 and 8).

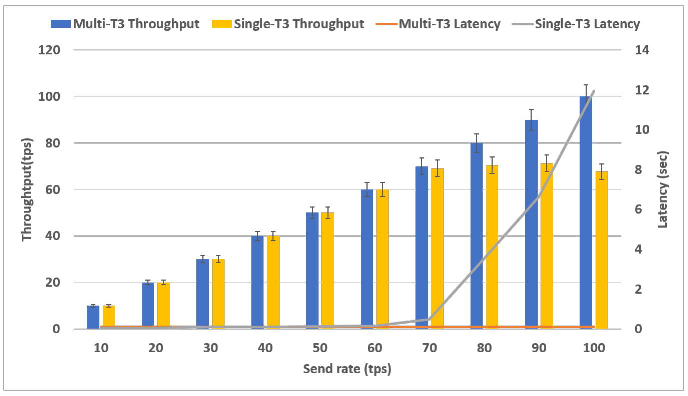
These two steps require *NotarySc* to construct the trade trail and perform trade validation and payment evaluation, which depends on the trail length. Hence, the execution time of these two components increases with the increase in trail length.

(3) Performance of System Contracts: Recall from section 5.2, TrailChain uses Layer 2 to record and manage the data ownership traceability for trading in the underlying marketplace networks. Herein, we evaluate the latency and throughput of the system contracts deployed in the Layer 2 blockchain network. Throughput is defined as the rate at which transactions are committed to the ledger. Latency is the time taken from when a node sends the transaction in the network to the time it is committed to the ledger. For Layer 2, we consider a base model of a solo miner node using clique as the consensus mechanism from predefined network models in Caliper. For the performance evaluations, we consider the read-only and state-changing transactions in both *TrackerSc* and *NotarySc* as they are not only frequently used but also their computational overhead is higher. To determine the scalability of TrailChain, we compared the performance in two scenarios: multi-T3 and single-T3. Multi-T3 corresponds to updating and reading trades from multiple $T3$ tables. While single-T3 corresponds to updating and reading trades from a single $T3$ table. Next, we present the throughput and latency evaluations for $Tx_{retrieveTrail}$ and $Tx_{updateT3}$ in *TrackerSc* and $Tx_{validateTrail}$ and $Tx_{evaluatepaymentShare}$ in *NotarySc*. We performed the evaluations by varying the send rate of transactions from 10 to 100 transactions per second (tps) for 1000 trades.

Read-only transactions: Web3 API provides a call function for local invocation of a contract function that does not broadcast or publish anything on the blockchain. $Tx_{retrieveTrail}$ and $Tx_{validateTrail}$ are read-only operations in *TrackerSc* and *NotarySc*, respectively, for reading and constructing the trade trail for a given data owner. The call transactions are synchronous, and the return value of the contract function is returned immediately. The performance results for $Tx_{retrieveTrail}$ and $Tx_{validateTrail}$ are shown in Fig. 5.20. As expected, the throughput of both read transactions increases linearly with an increase in send rate, while latency remains low for the multi-T3 scenario. However, in the single-T3 case, the throughput starts saturating at 70tps while the latency starts increasing. Recall



(a)

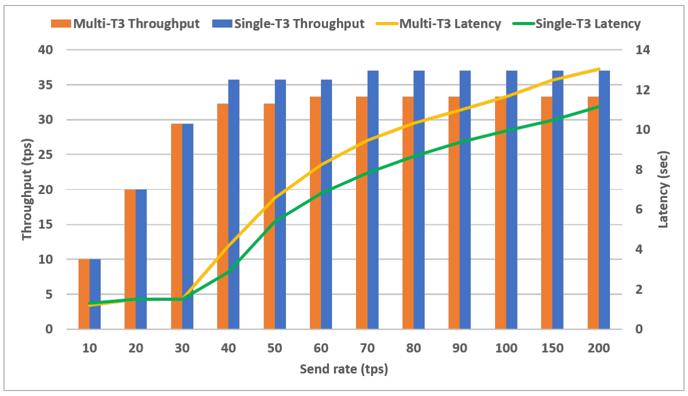


(b)

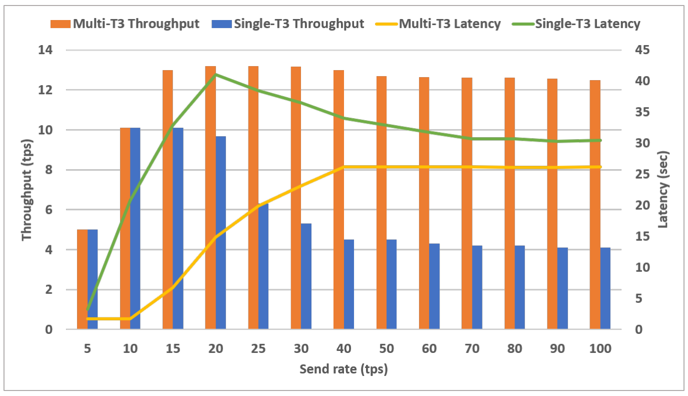
Figure 5.20: Throughput and latency for (a) $Tx_{retrieveTrail}$ (b) $Tx_{validateTrail}$

from section 5.3.5 that to retrieve or validate the trail, the trade trail is constructed using $T3$. In this experiment, a trail length for each read request is selected in the range of (1,1000) with uniform distribution. With the increase in send rate, the number of call transactions proliferates, and the node fails to construct the trails for some of the requests; hence, transactions start failing with a timeout error. Note that the throughput and latency of $Tx_{validateTrail}$ is slightly more than that of $Tx_{retrieveTrail}$. Contrary to *TrackerSc*, *NotarySc* uses an additional logic to validate the constructed trail leading to a slight increase in the metrics.

State-changing transactions: These transactions are broadcast to the network, processed by miners, and, if valid, are published on the blockchain. We consider $Tx_{updateT3}$ and $Tx_{evaluatepaymentShare}$ for performance analysis as these transactions perform write-operations



(a)



(b)

Figure 5.21: Throughput and latency for (a) $Tx_{updateT3}$ (b) $Tx_{evaluatepaymentShare}$

on Layer 2 that will update the state of the blockchain and consume Ether. Fig. 5.21a shows the performance results of $Tx_{updateT3}$. It is observed that in the multi-T3 scenario, throughput increases linearly with send rate. Throughput starts saturating at 33.33tps when the send rate approaches 40tps. A similar trend is observed in single-T3, with the throughput saturating at 37tps at the same transaction load. Furthermore, latency remains low (1.5s) and then starts increasing from send rate of 40tps. This is due to the congestion caused by the rapid growth in the number of transactions waiting in the execution and validation queues, which affects the commit latency. Before the saturation point, the throughput and latency for both Single-T3 and Multi-T3 are almost the same. However, after the saturation point, it is observed that the update in Single-T3 is faster than Multi-T3, where the latency of Single-T3 is lower, and throughput is higher than Multi-T3. This trend is similar to the bulk-insert operation in SQL wherein a multi-row insertion in one database is faster than inserting the same number of single-row in multiple databases. This difference in overhead is due to the extra time spent establishing connections with multiple databases rather than connecting to a single database. Similarly, inserting a single element in multiple tables in multi-T3 causes a higher latency than inserting multiple elements in a single table in single-T3.

$Tx_{evaluatepaymentShare}$ is the most expensive transaction in TrailChain as it involves three operations to provide a trusted payment share among the data owners. The throughput and latency comparison of $Tx_{evaluatepaymentShare}$ in single-T3 and multi-T3 are shown in Fig. 5.21b. In the case of multi-T3, the throughput increases linearly till send rate is 10tps and approaches to maximum performance point of 13.2tps for the send rate of 20-30tps. However, with a further increase in the send rate from this point, the throughput starts falling and settles to 12.5tps. The transaction latency of multi-T3 remains almost constant till the send rate is 10tps, and an increase is observed till 40tps. This is also due to the increase in the commit latency that is caused by the congestion of the number of transactions growing in the execution and validation queues. However, with a further increase in the send rate from 40tps, the latency saturates to 26sec. This is because, from 40tps, many transactions are reported as failed by the Caliper due to the high validation time of successful transactions and high waiting time of remaining transactions in the queue. Hence, the number of successful transactions starts decreasing. Caliper calculates the throughput and latency only for successful transactions [https://github.com/hyperledger/caliper]. Therefore, with the decrease and saturation in the number of successful transactions on increasing send rate beyond 40tps, we observe that the throughput and latency also get saturated. This implies that the system under test has reached its saturation point, and a further increase in send rate will not impact the performance. In the case of single-T3, we observed a similar trend for throughout that approaches the maximum performance point of 10tps for send rate of 15tps and then starts decreasing and settles at 4.1tps. However, in single-T3, we observe that even for a lower send rate of 5tps, the latency is higher compared to multi-T3. This can be explained by the fact that in the single-T3 scenario, a single $T3$ has 1000 trade elements, while in the multi-T3 case, each $T3$ has a single trade element. Therefore, $NotarySc$ takes more time to construct and validate the trade trail in single-T3 than in multi-T3. This increases the validation time for the single-T3 case. The latency reaches its peak of 41sec with the send rate at 20tps. Further increase in the send rate decreases the latency, which finally settles at 30sec. This is due to the decrease in the number of successful transactions as explained earlier.

Recall from section 5.2 that *TrackerSc* only manages the data storage-related functions in Layer 2, while the business logic related to trail validation and payment share evaluation is handled by *NotarySc*. Therefore, with the increase in the validation time of the transactions, the commit time of $Tx_{evaluatepaymentShare}$ is more than that of $Tx_{updateT3}$. Hence, the performance of $Tx_{updateT3}$ is better than $Tx_{evaluatepaymentShare}$ in both multi-T3 and single-T3 scenarios.

5.4.3 Security Analysis

In this section, we discuss the critical attacks that can be implemented by malicious entities including providers, data resellers, digital notary nodes and external attackers, and argue how TrailChain can provide resilience against all of them.

Malicious provider: In TrailChain, all providers are required to register their data in the marketplace before selling it. An actor cannot sell anything that is not registered in the marketplace. However, a malicious provider may try to register some fake/bogus/inauthentic data in the marketplace. Our system can detect such an attempt when the provider executes the data ownership registration process. The watermarking module embeds the watermark in the data generated from the IoT device after verifying that the device is registered and in possession of the provider, as explained in 5.3.4. The watermark will not be embedded if the data is not generated from the stated device. Hence, a malicious provider cannot register bogus/fake/inauthentic data in the marketplace.

Malicious data reseller: A data reseller can be involved in the following three malicious activities. In the first instance, a dishonest data reseller may try to subvert payment of the resell shares for purchased data D by not executing the data verification process. In this case, the data reseller will not be listed as an owner in the $T3(D)$. Moreover, suppose they resells the data to an honest buyer who executes the data verification process. In that case, the malicious data reseller will be automatically identified as an illegal reseller by the *NotarySc* as discussed in section 5.3.5. In the second instance, when the malicious data reseller tries to falsely claim ownership of the data by registering it under their name, the watermarking module will detect the already embedded watermark, and registration will fail. In the third instance, a malicious data reseller attempts to sell the data by modifying it. When they tries to register the ownership of the modified data, the execution of DORP will fail, as the data reseller will not be able to prove that the data is generated by a registered device that is owned by him. Moreover, TrailChain employs a backward watermark-chain scheme that will chain all batches using a watermark. Any change or tampering with the original data is thus easily detectable using this technique. A dishonest data reseller can be penalized by reducing their reputation or revoking their participation in the marketplace. These accountability mechanisms will be explored in our future work.

Malicious digital notary node: The key involvement of the digital notary node is in the data ownership registration protocol and data verification process, where they can act maliciously and degrade the effectiveness of the ownership traceability mechanism. Moreover, a malevolent digital notary can attempt to obtain the data content causing privacy or security risk to the provider. TrailChain prevents such attacks, as explained below.

- During the data ownership registration process, a dishonest digital notary can potentially change the tracking identifier TID' embedded in the data. Corresponding to an invalid tracking identifier, no $T3(D)$ exists in the data ownership management layer. Hence, genuine data will be regarded as invalid/fake data. TrailChain prevents this attack by enabling trusted cross-layer information exchange between the data ownership management network and marketplace network, wherein such information transfers are accompanied by attestation-based proof from the subset of digital notaries along with the tracking identifier. Hence, such malicious activity of modifying the tracking identifier TID by a dishonest digital notary can be easily detected and penalized.

- During the data verification process, a malicious digital notary extracts the watermark from the data and intentionally sends a different watermark to *NotarySc*. When *NotarySc* executes the trade lineage verification algorithm, the fallacious watermark or TID' will fail to identify trade records in the data ownership management layer and provide incorrect verification results. To prevent this attack, a marketplace network can register additional digital notaries. Consequently, instead of relying on a single digital notary, a buyer can send their DVP packets to multiple digital notaries for data verification. This minimizes the potential tampering of the extracted watermark by a dishonest digital notary.
- A malicious digital notary tries to obtain data using illegitimate means that can raise security or privacy issues for the provider. Since data transfer in TrailChain happens directly between data seller and buyer using TLS, the likelihood of such an attack is low. Furthermore, the buyer shares a small subset of data batches with a digital notary during DVP to validate the trade lineage and generate PoA, as explained in section 5.3.5. Since only a limited number of samples are shared, it will have a low impact on the security or privacy of the provider. However, even if the digital notary obtains the content of the data by any illegitimate means and wishes to sell the data, such unauthorized selling can be detected by TrailChain.

Colluding digital notary: Malicious digital notaries can collude to corrupt the cross-layer information transfer from TrailChain. This attack can be mitigated by employing a large number of validators in the data ownership management network. Furthermore, to generate proof for trusted information transfer, attestation by the signatures of more than 2/3 of the digital notaries can be used as in [256]. This will ensure better system security as compared to a pure majority (51%).

Device Impersonation: An attacker may try to impersonate a device to register fake/illegitimate data. However, this attack requires the attacker to have access to the actual physical device as the data registration process validates the source of data using the device's PUF, a hardware fingerprint as discussed in section 5.3.4. The likelihood of success is thus very low, and hence, TrailChain exhibits high resilience against device impersonation attacks.

Actor impersonation: An external attacker can generate fake transactions by trying to impersonate a legitimate actor. An attacker may sign the transactions by either stealing the victim's private key or using a different private key. In the former case, the attacker will require to access the personal storage of the victim where these keys are stored, which is highly difficult. In comparison, the system can easily detect the later case. This is because all transactions are signed using the actors' private keys, the corresponding public keys known to the system and used to verify the transactions. Therefore, the likelihood of success of this attack is very low.

Denial of Service: An adversary may make the digital notary unavailable by sending a large number of fake requests for DORP and TLV processes. DOS protection can be built into the digital notary using thresholding methods [257] in which incoming transaction requests from such malicious senders can be blocked and banned from the system if they generate transactions above a certain threshold. However, it can still lead to short-term DOS, whose effect can be further mitigated by adding redundant digital notaries in the marketplace.

5.5 Discussion

The joint use of watermarking techniques and blockchain technology in TrailChain ensures integrity, transparency and availability of data ownership information in a trustless environment to prove ownership and enable ownership traceability in data marketplace, respectively. A comparison of the most relevant works [131, 137, 139, 143] with TrailChain is summarized in Table 5.2. This comparison demonstrates several benefits offered by TrailChain in overcoming the challenges mentioned in section 5.1. (1) Data authenticity: TrailChain uses a PUF-based data ownership registration process that enhances the effec

Table 5.2: A Comparative summary of TrailChain wrt relevant works.

Metrics	**[139]**	**[137]**	**[131]**	**[143]**	**TrailChain**
Data type	Big data	IoT data	Image	IoT data	IoT data
Blockchain	Ethereum	NA	Ethereum	NA	Ethereum
Watermark	✓	✓	✓	✗	✓
Data authenticity	✗	✗	✗	✓	✓
Trusted trade trail	✓	✓	✓	✗	✓
Automatic reselling detection	✗	✗	✗	✓	✓
Fair distribution of resell payment	✗	✗	✗	✓	✓
Security	✗	✗	✗	✗	✓
Multi-system network	✗	✓	✗	✗	✓
Performance evaluation	✗	✗	✗	✗	✓

tiveness and usability of the marketplace by ensuring that only genuine data is registered and traded in the system. TrailChain also provides a data verification mechanism for buyers to validate the authenticity of the purchased data. (2) Trusted trade trail: TrailChain uses system contracts to trace data ownership and provide a transparent, immutable audit trail for data movement across multiple marketplace networks. (3) Automatic reselling Detection: Our model automatically detects data reselling and identifies illegitimate data trading and double-buying scenarios. (4) Fair distribution of resell payment: TrailChain adopts a payment-share channel to allow fair and trusted distribution of resell payments among the owners in the trail. (5) Security: TrailChain provides resiliency against various security threats, minimizing the overall risks in the data ownership traceability mechanism. (6) Multi-system networks: TrailChain provides an effective data ownership traceability in the multi-marketplace data trading scenario. (7) Performance evaluation: The experimental evaluation of TrailChain demonstrates its effectiveness in detecting undisclosed reselling, identifying unauthorized data reselling, and fairly distributing the revenue among the data owners at marginal overhead.

Our approach uses a common token management layer for simplifying payment settlement among data owners in different blockchain-based marketplace networks. More complex scenarios that require asset exchange in different blockchain networks can be implemented using techniques like Hash Time Locked Contracts (HTLC) [258] or atomic swaps [259]. In the future, we will consider incorporating these techniques into our architecture to enable a wider spectrum of applications, including asset and information transfers.

Another limitation is that the robustness and efficiency of TrailChain depend on the underlying coupling of the embedded watermark and the data. To ensure data integrity and tamper detection, TrailChain adopts a backward watermark-chaining technique, wherein each data batch is linked with the previous batch through the watermark as explained in section 5.2.2, similar to blockchain. Hence, making it difficult for an attacker to remove the embedded watermark or make it undetectable while keeping the data useful. To prevent forgery or modification attacks, a robust watermark technique [260] can be used. Moreover, to demonstrate the generality of the traded data in the TrailChain, we presented a proof of concept for GPS data samples. However, we expect TrailChain to work even better for trading multimedia data (image, video) with more advanced watermarking techniques [261, 262] that provide stronger coupling between the data and watermark.

The quantification of throughput and latency of transactions depends on various factors [263] such as the underlying consensus mechanism, blockchain platform used, execution environment, block time etc. However, the trends of throughput and latency will likely remain the same as they depend on the underlying logic of smart contracts. In this chapter, we used the Ethereum blockchain to illustrate the feasibility of our architecture and demonstrate that our design works even on a permission-less blockchain. However, other blockchain platforms such as Hyperledger Fabric, Besu or Corda with other consensus mechanisms (iBFT, raft, PBFT) could be considered that can further improve the system's performance for varying numbers of transactions [264]. Furthermore, since the simulation of POC is performed on a single machine, the end-to-end execution time analysis does not incorporate network delay and therefore, the latency due to cross-layer transactions or message exchange between two entities is considered to be negligible.

5.6 Chapter Summary

In this chapter, we proposed a data ownership management and traceability framework, TrailChain, which aims to provide a secure, immutable and trusted trade trail to trace the data movement across multiple decentralized marketplace networks. TrailChain uses a blockchain-based data ownership management layer that keeps records of the ownership changes when the data is traded among the marketplace participants. The framework also provides a PUF-based data ownership registration process that ensures the reliability of the origin of data. Additionally, it provides a data verification mechanism for buyers to validate the authenticity of the bought data. TrailChain achieves automation and efficiency using smart contracts that enable resell detection and ensure that the revenue is properly and fairly shared among all the data owners in the trail. TrailChain is agnostic of the underlying marketplace design and can be implemented for existing designs without requiring many changes. We performed a qualitative security analysis to show that the approach is immune to several common attacks. We implemented a proof-of-concept implementation on Ethereum and demonstrated its feasibility in terms of gas costs and execution time. Lastly, we performed an experimental evaluation on Hyperledger Caliper to demonstrate its effectiveness in terms of throughput and latency.

Chapter 6

Conclusion and Future Directions

In this thesis, we explored the adoption of blockchain to achieve autonomy, efficiency, privacy and traceability in the decentralized IoT data marketplace. First, we presented an extensive literature review to understand the state-of-the-art blockchain-based IoT data marketplace and the associated challenges from the perspective of ensuring autonomy, efficiency, privacy and traceability. Second, we presented MartChain, an autonomous and efficient marketplace framework, to address the challenges of scalability, repudiation and the resource-constrained nature of IoT devices. Third, we presented KYBChain, a privacy-aware marketplace framework, to address the privacy concerns of providers resulting from sharing their sensitive IoT data with buyers about whom they may not have any prior information or knowledge. Finally, we presented TrailChain, a data ownership traceability framework, to address the complex issues of ambiguous data ownership spanning multiple marketplace systems, undisclosed reselling and lack of a mechanism to verify the authenticity of purchased data.

In the next section, we summarize the contributions made in each chapter, followed by potential research directions in section 6.2.

6.1 Summary of Contributions

We proposed MartChain in Chapter 3, a fully functional and effective decentralized marketplace for trading IoT data stream in real-time from resource-constrained IoT devices. The framework design is based on the autonomy and efficiency requirements of the data marketplace. The design enforces a holistic model equipped with all essential components of the marketplace, such as discovery and selection, data agreement management, price model and reputation mechanism. We outlined the participant's roles in MartChain and the interaction among them. We also presented the functionalities of four smart contracts (*SubscriptionSc*, *RegisterSc*, *PriceSc*, and *ReputationSc*) and their associated methods to achieve trustful automated data trading. The framework also provides a simple and dynamic way of pricing the data based on the competitors' prices in the market and encourages providers to sell high-quality data and value-added features in return for greater incentives. To provide resilience to various security attacks, we presented a reputation mechanism that uses the trading history of entities in tuning the reputation score and penalizes them for being dishonest. An EDSA mechanism underpinned MartChain to ensure that the computational constraints of the provider's IoT devices, such as battery energy, processing capacity, and communication bandwidth, are considered in determining which data requests a provider can service. This represents a multi-objective optimization problem that incorporates the computational constraints of the provider's IoT devices, requests from buyers, and the need to maximize profits and value for the provider and buyer, respectively. We implemented the key MartChain components as smart contracts on the Ethereum blockchain. We evaluated the gas consumption and cost incurred for interacting with MartChain. Furthermore, we formulated the EDSA problem in MATLAB and simulated various scenarios to analyze the impact of the buyer's requests on the battery drainage of the provider's devices. The simulation result shows that our approach is viable and benefits the provider and buyer by creating an autonomous and efficient real-time data trading model.

In Chapter 4, we proposed KYBChain, a know-your-buyer marketplace framework integrated with a privacy rating system. The contribution of this chapter is two-fold. First, we proposed a privacy rating model to address the privacy concerns of providers related to non-compliance risk, data accumulation risk and data leakage risk. Corresponding to each risk, we maintained three profiles of the buyer, i.e. practice, purchase and leakage, based on their different characteristics in the marketplace. We identified several privacy elements to model these profiles. These identified privacy elements are further used to develop rubrics for scoring the magnitude of risk associated with each profile. We then developed a methodology to convert these risk scores to the buyer's respective ratings. Second, we designed a blockchain-based marketplace framework embedded with a standalone rating management component, the KYB-module, that records all buyers' profiles. KYB-module comprises smart contracts that automatically map the buyer's activities with their profile. Moreover, we presented details of the methods and transactions implemented in these smart contracts and their interaction with the marketplace components during different stages of KYBChain. We conducted multiple simulations to show the utility of privacy ratings based on three providers' privacy dispositions: unconcerned, fundamentalist and pragmatist. We developed a proof-of-concept implementation of KYBChain in a private Ethereum network and reported experimental results regarding gas consumption, throughput and latency using Hyperledger Caliper. Our results revealed that compared to a marketplace that does not incorporate a privacy rating system, the overheads introduced by our mechanism are insignificant relative to its privacy gains.

Finally, in Chapter 5, we proposed TrailChain, a data ownership management and traceability framework, which aims to provide a secure, immutable and trusted trade trail to trace the data movement across multiple decentralized marketplace networks. We devised a PUF-based data ownership registration process that allows data producers to register the ownership of original data and guarantees that a genuine device has generated the data. TrailChain provided a data verification mechanism for buyers to validate the authenticity of the bought data. We also proposed a fair resell payment sharing scheme that allows trusted, protected and automated sharing of resell revenue among the data owners in the trade trail. Finally, we developed a prototype implementation of TrailChain on Ethereum and reported detailed experimental results regarding gas consumption and end-to-end execution time. We performed an experimental evaluation on Hyperledger

Caliper to show its effectiveness in terms of throughput and latency. We also performed a qualitative security analysis of TrailChain's resilience to various malicious activities. Simulations demonstrated that our method detected undisclosed reselling within the same marketplace and across different marketplaces. Besides, it identified whether or not the provider has authorized the reselling and fairly distributes the revenue among the data owners at marginal overhead.

6.2 Future Research Directions

The research presented in this dissertation augments novel contributions towards adopting blockchain in a decentralized IoT Data Marketplace. To conclude this chapter and the thesis, we present potential future research directions as described below:

The first area for future exploration is regulatory oversight of the marketplace for data trading. With its economical and societal benefits, data is a valuable asset for organizations, governments, individuals, etc. However, keeping private data and sensitive information safe is paramount with the ever-growing cyber security risks. Many lawmakers worldwide have introduced data protection laws to secure individuals' data privacy, availability, and integrity. Per a UN report [265], 137 out of 194 countries have implemented data protection and privacy legislation. However, these regional regulations are inconsistent and do not have significant international implications. Since the data marketplace network spans the globe, data can travel the world through this borderless network, making it challenging to apply data protection regulation evenly as citizens' data leaves that region. Therefore, considering regional data protection laws while designing a data marketplace to enable data transfer from one country to another would be an interesting future direction.

Another exciting research direction is the reliable data verification process for buyers. Compared to physical trading goods, verifying data against the buyer's requirements is difficult before buying them. A buyer can verify the data after buying it using our data verification process proposed in Chapter 5. However, data verification after buying reveals the data to the buyer, which may affect providers' security and privacy requirements. Moreover, since the value of data depends on its novelty, revealing data for verification may decrease its value. Existing approaches use trust mechanisms to establish trust that the provider will provide quality data per the buyer's requirements. However, trust in the data (the trading good) is still missing. It will be worth investigating new approaches for data verification addressing this privacy and trust trade-off. Cryptographic methods such as zero-knowledge proof that enable data verification without revealing the provider's identity will likely attract significant attention.

Another open area for future investigation is to reduce reliance on a central entity. For example, MartChain, discussed in Chapter 3, relied on a facilitator to provide various services such as search and discovery mechanisms and dispute resolution. In TrailChain presented in Chapter 5, digital notaries enable interoperability across ownership management, marketplace systems and payment layers. However, including such a central entity may lead to a certain level of monopoly, affect transparency, and centralize trust, making the purpose of using a blockchain futile. Therefore, it is critical to reduce or remove the dependency on the central entity to achieve decentralized solutions.

Another potential direction for future research is to acquire real-world data to simulate experiments. While the experiments conducted in this thesis are based on synthetic data, obtaining real data from industry or collaborations would be a valuable future direction to enhance the validity of the thesis's results and observations. This is an essential step towards demonstrating the effectiveness and practicality of the proposed decentralized IoT data marketplace in real-world scenarios.

Given the high volume of transactions, scalability limitations are expected to be a bottleneck for blockchain in data marketplace contexts. In MartChain (see Chapter 3), we employed geographically distributed facilitators to address the issue of scalability. We delegated two marketplace operations to the facilitators, including listing data offerings from myriads of devices and searching and matching offerings based on buyers' queries. While we employed blockchain to manage only trade-related transactions. However, the transaction volume from millions of users on the blockchain can significantly affect the system's performance, including latency, throughput and cost. It will be worth exploring the cluster shard [266] approach to address the scalability constraints of blockchain for massive transaction volume. In the cluster shard approach, instead of using a single ledger to record all the transactions, the entire ledger can be split into smaller chunks, known as shards, that can operate independently to manage the transactions of its cluster.

Finally, applying machine learning (ML) in the IoT data marketplace for data pricing optimization is an interesting research direction. Pricing is used as an economic mechanism for revenue generation for providers. Optimizing prices to maximize providers' profits and avoid discouraging buyers from purchasing data has always been challenging. Existing studies used rule-based optimization methods wherein price rules are defined to determine the price-setting logic. Optimization tools automatically adjust the price based on these price rules. However, the traditional methods are time-consuming and ineffective, given the large volume of data that requires continuous evaluation. Combining two disruptive technologies, machine learning and blockchain, can be an effective solution compared to traditional approaches. ML algorithms can analyze large volumes of data, consider more variables and predict the price based on market dynamics. The immutable trade transactions stored in the blockchain can be fed to ML algorithms to develop predictive models to calculate the price based on demand fluctuation. Developing an ML-based pricing model is an interesting future research direction that may make the data marketplace more attractive for broad adoption among the participants.

In summary, there is still much work to be done to realize the full capability of blockchain in the IoT data marketplace. Despite all these open challenges, blockchain, with its salient features of transparency, decentralization, trust, security, immutability and anonymity, will continue to hold a lot of potential for enabling efficient data democratization in the IoT data marketplace.

References

[1] C. Humby, “Data is the new oil,” *Proc. ANA Sr. Marketer’s Summit. Evanston, IL, USA*, vol. 1, 2006.

[2] K. Schwab, A. Marcus, J. Oyola, W. Hoffman, and M. Luzi, “Personal data: The emergence of a new asset class,” in *An Initiative of the World Economic Forum*, 2011.

[3] L. Dignan, “Iot devices to generate 79.4zb of data in 2025, says idc,” June 2019. [Online]. Available: https://www.zdnet.com/article/iot-devices-to-generate-79-4zb-of-data-in-2025-says-idc/

[4] A. Tahiliani, V. Hassija, V. Chamola, S. S. Kanhere, M. Guizani *et al.*, “Privacy-preserving and incentivized contact tracing for covid-19 using blockchain,” *IEEE Internet of Things Magazine*, vol. 4, no. 3, pp. 72–79, 2021.

[5] X. Krasniqi and E. Hajrizi, “Use of iot technology to drive the automotive industry from connected to full autonomous vehicles,” *IFAC-PapersOnLine*, vol. 49, no. 29, pp. 269–274, 2016.

[6] D. Pavithra and R. Balakrishnan, “Iot based monitoring and control system for home automation,” in *2015 global conference on communication technologies (GCCT)*. IEEE, 2015, pp. 169–173.

[7] L. Catarinucci, D. De Donno, L. Mainetti, L. Palano, L. Patrono, M. L. Stefanizzi, and L. Tarricone, “An iot-aware architecture for smart healthcare systems,” *IEEE internet of things journal*, vol. 2, no. 6, pp. 515–526, 2015.

[8] A. Spender, C. Bullen, L. Altmann-Richer, J. Cripps, R. Duffy, C. Falkous, M. Farrell, T. Horn, J. Wigzell, and W. Yeap, “Wearables and the internet of things: Considerations for the life and health insurance industry,” *British Actuarial Journal*, vol. 24, 2019.

[9] L. Georgios, S. Kerstin, and A. Theofylaktos, “Internet of things in the context of industry 4.0: An overview,” *International Journal of Entrepreneurial Knowledge*, 2019.

[10] M. M. Rathore, A. Ahmad, A. Paul, and S. Rho, “Urban planning and building smart cities based on the internet of things using big data analytics,” *Computer networks*, vol. 101, pp. 63–80, 2016.

[11] G. S. Ramachandran, R. Radhakrishnan, and B. Krishnamachari, “Towards a decentralized data marketplace for smart cities,” in *2018 IEEE International Smart Cities Conference (ISC2)*. IEEE, 2018, pp. 1–8.

[12] A. Turner, “How many smartphones are in the world?” 2022. [Online]. Available: https://www.bankmycell.com/blog/how-many-phones-are-in-the-world

[13] A. Javed, M. A. Shahid, M. Sharif, and M. Yasmin, “Energy consumption in mobile phones,” *International Journal of Computer Network and Information Security*, vol. 10, no. 12, p. 18, 2017.

[14] B. Heslop, “By 2030, each person will own 15 connected devices — here’s what that means for your business and content,” 2021. [Online]. Available: https://www.spiceworks.com/tech/iot/articles/by-2030-each-person-will-own-15-connected-devices-heres-what-that-means-for-your-business-and-content/

[15] S. C. Mukhopadhyay and N. K. Suryadevara, “Internet of things: Challenges and opportunities,” *Internet of Things*, pp. 1–17, 2014.

[16] J. Johnson, “What is dark data? the basics & the challenges,” July 2020. [Online]. Available: https://www.bmc.com/blogs/dark-data/

[17] T. Z. Yun, “The battle between targeted advertising & data privacy,” 2021. [Online]. Available: https://www.theedgemarkets.com/article/battle-between-targeted-advertising-data-privacy%C2%A0

[18] “Advertising revenue of google from 2001 to 2021,” 2022. [Online]. Available: https://www.statista.com/statistics/266249/advertising-revenue-of-google/

[19] S. Meredith, “Here’s everything you need to know about the cambridge analytica scandal,” 2021. [Online]. Available: https://www.cnbc.com/2018/03/21/facebook-cambridge-analytica-scandal-everything-you-need-to-know.html

[20] J. Cooper and A. James, “Challenges for database management in the internet of things,” *IETE Technical Review*, vol. 26, no. 5, pp. 320–329, 2009.

[21] T.-D. Cao, T.-V. Pham, Q.-H. Vu, H.-L. Truong, D.-H. Le, and S. Dustdar, "Marsa: A marketplace for realtime human sensing data," *ACM Transactions on Internet Technology (TOIT)*, vol. 16, no. 3, pp. 1–21, 2016.

[22] P. Treleaven, J. Barnett, A. Knight, and W. Serrano, "Real estate data marketplace," *AI and Ethics*, vol. 1, no. 4, pp. 445–462, 2021.

[23] A. Maynard and K. Bloor, "Trust and performance management in the medical marketplace," *Journal of the Royal Society of Medicine*, vol. 96, no. 11, pp. 532–539, 2003.

[24] K. Mišura and M. Žagar, "Data marketplace for internet of things," in *2016 International Conference on Smart Systems and Technologies (SST)*. IEEE, 2016, pp. 255–260.

[25] A. Bamrara, "Evaluating database security and cyber attacks: A relational approach," *The Journal of Internet Banking and Commerce*, vol. 20, no. 2, 2015.

[26] H.-Y. Paik, X. Xu, H. D. Bandara, S. U. Lee, and S. K. Lo, "Analysis of data management in blockchain-based systems: From architecture to governance," *Ieee Access*, vol. 7, pp. 186 091–186 107, 2019.

[27] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[28] D. Yaga, P. Mell, N. Roby, and K. Scarfone, "Blockchain technology overview," *arXiv preprint arXiv:1906.11078*, 2019.

[29] Y. Zhao, Y. Li, Q. Mu, B. Yang, and Y. Yu, "Secure pub-sub: Blockchain-based fair payment with reputation for reliable cyber physical systems," *IEEE Access*, vol. 6, pp. 12 295–12 303, 2018.

[30] G. F. Camilo, G. A. F. Rebello, L. A. C. de Souza, and O. C. M. Duarte, "Autavailchain: Automatic and secure data availability through blockchain," in *GLOBECOM 2020-2020 IEEE Global Communications Conference*. IEEE, 2020, pp. 1–6.

[31] 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, pp. 1–8.

[32] D. Liu, C. Huang, J. Ni, X. Lin, and X. S. Shen, "Blockchain-cloud transparent data marketing: Consortium management and fairness," *IEEE Transactions on Computers*, 2022.

[33] M. J. M. Chowdhury, M. S. Ferdous, K. Biswas, N. Chowdhury, A. Kayes, P. Watters, and A. Ng, "Trust modeling for blockchain-based wearable data market," in *2019 IEEE International Conference on Cloud Computing Technology and Science (CloudCom).* IEEE, 2019, pp. 411–417.

[34] M. Naz, F. A. Al-zahrani, R. Khalid, N. Javaid, A. M. Qamar, M. K. Afzal, and M. Shafiq, "A secure data sharing platform using blockchain and interplanetary file system," *Sustainability*, vol. 11, no. 24, p. 7054, 2019.

[35] Y. Xiao, N. Zhang, J. Li, W. Lou, and Y. T. Hou, "Privacyguard: Enforcing private data usage control with blockchain and attested off-chain contract execution," in *European Symposium on Research in Computer Security.* Springer, 2020, pp. 610–629.

[36] V. P. Ranganthan, R. Dantu, A. Paul, P. Mears, and K. Morozov, "A decentralized marketplace application on the ethereum blockchain," in *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC).* IEEE, 2018, pp. 90–97.

[37] K. R. Özyilmaz, M. Doğan, and A. Yurdakul, "Idmob: Iot data marketplace on blockchain," in *2018 crypto valley conference on blockchain technology (CVCBT).* IEEE, 2018, pp. 11–19.

[38] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 3–16.

[39] J. Siim, "Proof-of-stake," in *Research seminar in cryptography*, 2017.

[40] K. Driscoll, B. Hall, H. Sivencrona, and P. Zumsteg, "Byzantine fault tolerance, from theory to reality," in *International Conference on Computer Safety, Reliability, and Security.* Springer, 2003, pp. 235–248.

[41] A. Reyna, C. Martín, J. Chen, E. Soler, and M. Díaz, "On blockchain and its integration with iot. challenges and opportunities," *Future generation computer systems*, vol. 88, pp. 173–190, 2018.

[42] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, vol. 3, no. 37, pp. 2–1, 2014.

[43] "IOTA," 2022. [Online]. Available: https://www.iota.org/

[44] IOTA Foundation, "Introducing masked authenticated messaging," 2017. [Online]. Available: https://blog.iota.org/introducing-masked-authenticated-messaging-e55c1822d50e/

[45] "Data Marketplace," 2022. [Online]. Available: https://wiki.iota.org/blueprints/data-marketplace/overview

[46] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich *et al.*, "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the thirteenth EuroSys conference*, 2018, pp. 1–15.

[47] R. G. Brown, J. Carlyle, I. Grigg, and M. Hearn, "Corda: an introduction," *R3 CEV, August*, vol. 1, no. 15, p. 14, 2016.

[48] J. Morgan, "Quorum whitepaper," *New York: JP Morgan Chase*, 2016.

[49] G. Greenspan *et al.*, "Multichain private blockchain-white paper," *URl: http://www. multichain. com/download/MultiChain-White-Paper. pdf*, vol. 85, 2015.

[50] W. J. Luther, "Bitcoin and the future of digital payments," *The Independent Review*, vol. 20, no. 3, pp. 397–404, 2016.

[51] B. Krishnamachari, J. Power, C. Shahabi, and S. H. Kim, "Iot marketplace: A data and api market for iot devices," *University of Southern California: Los Angeles, CA, USA*, 2017.

[52] C. Perera, M. Barhamgi, S. De, T. Baarslag, M. Vecchio, and K.-K. R. Choo, "Designing the Sensing as a Service Ecosystem for the Internet of Things," *IEEE Internet of Things Magazine*, vol. 1, no. 2, pp. 18–23, 2018.

[53] 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)*. IEEE, 2019, pp. 339–346.

[54] A. Suliman, Z. Husain, M. Abououf, M. Alblooshi, and K. Salah, "Monetization of iot data using smart contracts," *IET Networks*, vol. 8, no. 1, pp. 32–37, 2019.

[55] Z. Huang, X. Su, Y. Zhang, C. Shi, H. Zhang, and L. Xie, "A decentralized solution for iot data trusted exchange based-on blockchain," in *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*. IEEE, 2017, pp. 1180–1184.

[56] L. Mikkelsen, K. Mortensen, H. Rasmussen, H.-P. Schwefel, and T. Madsen, "Realization and evaluation of marketplace functionalities using ethereum blockchain," in *2018 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC)*. IEEE, 2018, pp. 47–52.

[57] "Databroker," 2022. [Online]. Available: https://www.databroker.global/

[58] "Moeco," 2022. [Online]. Available: https://moeco.io/

[59] "Streamr," 2022. [Online]. Available: https://streamr.network/

[60] A. Javaid, M. Zahid, I. Ali, R. J. U. H. Khan, Z. Noshad, and N. Javaid, "Reputation system for iot data monetization using blockchain," in *International Conference on Broadband and Wireless Computing, Communication and Applications.* Springer, 2019, pp. 173–184.

[61] P. Tzianos, G. Pipelidis, and N. Tsiamitros, "Hermes: An open and transparent marketplace for iot sensor data over distributed ledgers," in *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC).* IEEE, 2019, pp. 167–170.

[62] P. Banerjee and S. Ruj, "Blockchain enabled data marketplace–design and challenges," *arXiv preprint arXiv:1811.11462*, 2018.

[63] M. Travizano, C. Sarraute, G. Ajzenman, and M. Minnoni, "Wibson: A decentralized data marketplace," *arXiv preprint arXiv:1812.09966*, 2018.

[64] H. T. T. Truong, M. Almeida, G. Karame, and C. Soriente, "Towards secure and decentralized sharing of iot data," in *2019 IEEE International Conference on Blockchain (Blockchain).* IEEE, 2019, pp. 176–183.

[65] H. Shafagh, L. Burkhalter, A. Hithnawi, and S. Duquennoy, "Towards blockchain-based auditable storage and sharing of iot data," in *Proceedings of the 2017 on cloud computing security workshop*, 2017, pp. 45–50.

[66] "Datapace," 2022. [Online]. Available: https://open-ecosystem.org/partners/datapace

[67] "Mainflux IoT platform," 2022. [Online]. Available: https://mainflux.com/cloud.html

[68] "Datum," 2022. [Online]. Available: https://datum.org/

[69] N. A. Koul, A. Sabrina, and J.-h. Morin, "Agreements framework for data market ecosystem," *WISP Proceedings*, 2018.

[70] R. Radhakrishnan, G. S. Ramachandran, and B. Krishnamachari, "Sdpp: Streaming data payment protocol for data economy," in *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC).* IEEE, 2019, pp. 17–18.

[71] M. Zichichi, L. Serena, S. Ferretti, and G. D'Angelo, "Towards decentralized complex queries over distributed ledgers: a data marketplace use-case," in *2021 International Conference on Computer Communications and Networks (ICCCN).* IEEE, 2021, pp. 1–6.

[72] Y.-N. Li, X. Feng, J. Xie, H. Feng, Z. Guan, and Q. Wu, "A decentralized and secure blockchain platform for open fair data trading," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 7, p. e5578, 2020.

[73] A. Manzoor, A. Braeken, S. S. Kanhere, M. Ylianttila, and M. Liyanage, "Proxy re-encryption enabled secure and anonymous iot data sharing platform based on blockchain," *Journal of Network and Computer Applications*, vol. 176, p. 102917, 2021.

[74] N. Hynes, D. Dao, D. Yan, R. Cheng, and D. Song, "A demonstration of sterling: a privacy-preserving data marketplace," *Proceedings of the VLDB Endowment*, vol. 11, no. 12, pp. 2086–2089, 2018.

[75] X. Zheng, "Data trading with differential privacy in data market," in *Proceedings of 2020 the 6th International Conference on Computing and Data Engineering*, 2020, pp. 112–115.

[76] M. Zichichi, M. Contu, S. Ferretti, V. Rodríguez-Doncel *et al.*, "Ensuring personal data anonymity in data marketplaces through sensing-as-a-service and distributed ledger technologies," in *Proceedings of the 3rd Distributed Ledger Technology Workshop.* ITA, 2020.

[77] Y. Zhao, Y. Yu, Y. Li, G. Han, and X. Du, "Machine learning based privacy-preserving fair data trading in big data market," *Information Sciences*, vol. 478, pp. 449–460, 2019.

[78] W. Dai, C. Dai, K.-K. R. Choo, C. Cui, D. Zou, and H. Jin, "Sdte: A secure blockchain-based data trading ecosystem," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 725–737, 2019.

[79] V. Koutsos, D. Papadopoulos, D. Chatzopoulos, S. Tarkoma, and P. Hui, "Agora: a privacy-aware data marketplace," *IEEE Transactions on Dependable and Secure Computing*, 2021.

[80] Latham & Watkins LLP, "Privacy Enhancing Technologies - A Panacea for Data Protection Compliance?" 2022. [Online]. Available: https://www.globalprivacyblog.com/privacy/privacy-enhancing-technologies-a-panacea-for-data-protection-compliance/

[81] A. Aggarwal, W. Asif, H. Azam, M. Markovic, M. Rajarajan, and P. Edwards, "User privacy risk analysis for the internet of things," in *2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS).* IEEE, 2019, pp. 259–264.

[82] V. Shivraj, M. Rajan, and P. Balamuralidhar, "A graph theory based generic risk assessment framework for internet of things (iot)," in *2017 IEEE international conference on advanced networks and telecommunications systems (ANTS).* IEEE, 2017, pp. 1–6.

[83] M. Markovic, W. Asif, D. Corsar, N. Jacobs, P. Edwards, M. Rajarajan, and C. Cottrill, "Towards automated privacy risk assessments in iot systems," in *Proceedings of the 5th Workshop on Middleware and Applications for the Internet of Things*, 2018, pp. 15–18.

[84] M. Elkhodr, B. Alsinglawi, and M. Alshehri, "A privacy risk assessment for the internet of things in healthcare," in *Applications of intelligent technologies in healthcare.* Springer, 2019, pp. 47–54.

[85] I. Psychoula, L. Chen, and O. Amft, "Privacy risk awareness in wearables and the internet of things," *IEEE Pervasive Computing*, vol. 19, no. 3, pp. 60–66, 2020.

[86] M. Tavakolan and I. A. Faridi, "Applying privacy-aware policies in iot devices using privacy metrics," in *2020 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI).* IEEE, 2020, pp. 1–5.

[87] B. Thuraisingham, M. Kantarcioglu, E. Bertino, J. Z. Bakdash, and M. Fernandez, "Towards a privacy-aware quantified self data management framework," in *Proceedings of the 23nd ACM on Symposium on Access Control Models and Technologies*, 2018, pp. 173–184.

[88] J. Iyilade and J. Vassileva, "A framework for privacy-aware user data trading," in *International Conference on User Modeling, Adaptation, and Personalization.* Springer, 2013, pp. 310–317.

[89] S. M. Eckartz, W. J. Hofman, and A. F. V. Veenstra, "A decision model for data sharing," in *International conference on electronic government.* Springer, 2014, pp. 253–264.

[90] H. Wimmer, V. Y. Yoon, and V. Sugumaran, "A multi-agent system to support evidence based medicine and clinical decision making via data sharing and data privacy," *Decision Support Systems*, vol. 88, pp. 51–66, 2016.

[91] G. F. Camilo, G. A. F. Rebello, L. A. C. de Souza, and O. C. M. Duarte, "A secure personal-data trading system based on blockchain, trust, and reputation," in *2020 IEEE International conference on blockchain (blockchain).* IEEE, 2020, pp. 379–384.

[92] G. D. Putra, V. Dedeoglu, S. S. Kanhere, and R. Jurdak, "Trust management in decentralized iot access control system," in *2020 IEEE international conference on blockchain and cryptocurrency (ICBC).* IEEE, 2020, pp. 1–9.

[93] J. M. Wills, D. B. Schmidt, F. Pillo-Blocka, and G. Cairns, "Exploring global consumer attitudes toward nutrition information on food labels," *Nutrition reviews*, vol. 67, no. suppl_1, pp. S102–S106, 2009.

[94] A. Yezioro and I. G. Capeluto, “Energy rating of buildings to promote energy-conscious design in israel,” *Buildings*, vol. 11, no. 2, p. 59, 2021.

[95] “Apple privacy label,” 2022. [Online]. Available: https://www.apple.com/privacy/labels/

[96] “Google data safety,” 2022. [Online]. Available: https://support.google.com/googleplay/android-developer/answer/10787469?hl=en

[97] Y. Shen and P.-A. Vervier, “Iot security and privacy labels,” in *Annual Privacy Forum.* Springer, 2019, pp. 136–147.

[98] P. Emami-Naeini, Y. Agarwal, L. F. Cranor, and H. Hibshi, “Ask the experts: What should be on an iot privacy and security label?” in *2020 IEEE Symposium on Security and Privacy (SP).* IEEE, 2020, pp. 447–464.

[99] P. G. Kelley, J. Bresee, L. F. Cranor, and R. W. Reeder, “A" nutrition label" for privacy,” in *Proceedings of the 5th Symposium on Usable Privacy and Security*, 2009, pp. 1–12.

[100] S. Allana and S. Chawla, “Childshield: A rating system for assessing privacy and security of internet of toys,” *Telematics and Informatics*, vol. 56, p. 101477, 2021.

[101] K. Liu and E. Terzi, “A framework for computing the privacy scores of users in online social networks,” *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 5, no. 1, pp. 1–30, 2010.

[102] E. Aghasian, S. Garg, L. Gao, S. Yu, and J. Montgomery, “Scoring users’ privacy disclosure across multiple online social networks,” *IEEE access*, vol. 5, pp. 13 118–13 130, 2017.

[103] M. Maass, P. Wichmann, H. Pridöhl, and D. Herrmann, “Privacyscore: Improving privacy and security via crowd-sourced benchmarks of websites,” in *Annual Privacy Forum.* Springer, 2017, pp. 178–191.

[104] J. S. Montgomery, “A privacy risk scoring framework for mobile applications and platforms,” Ph.D. dissertation, BYU ScholarsArchive, 2014.

[105] M. Shaker, F. Shams Aliee, and R. Fotohi, “Online rating system development using blockchain-based distributed ledger technology,” *Wireless Networks*, vol. 27, no. 3, pp. 1715–1737, 2021.

[106] Y. M. Arif, H. Nurhayati, S. Harini, S. M. S. Nugroho, and M. Hariadi, “Decentralized tourism destinations rating system using 6astd framework and blockchain,” in *2020 International Conference on Smart Technology and Applications (ICoSTA).* IEEE, 2020, pp. 1–6.

[107] G. Maragatham *et al.*, "Movie rating system based on blockchain," in *2021 International Conference on Computer Communication and Informatics (ICCCI).* IEEE, 2021, pp. 1–3.

[108] N. Jain, T. Agrawal, P. Goyal, and V. Hassija, "A blockchain-based distributed network for secure credit scoring," in *2019 5th international conference on signal processing, computing and control (ISPCC).* IEEE, 2019, pp. 306–312.

[109] S. Chakraborty, S. Aich, S. J. Seong, and H.-C. Kim, "A blockchain based credit analysis framework for efficient financial systems," in *2019 21st International Conference on Advanced Communication Technology (ICACT).* IEEE, 2019, pp. 56–60.

[110] V. Hassija, G. Bansal, V. Chamola, N. Kumar, and M. Guizani, "Secure lending: Blockchain and prospect theory-based decentralized credit scoring model," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 2566–2575, 2020.

[111] M. S. Iftikhar, Z. A. Khan, Z. Noshad, A. Khalid, and N. Javaid, "Reliable services from service providers based on the ratings of iot devices using blockchain," in *2019 Sixth HCT Information Technology Trends (ITT).* IEEE, 2019, pp. 73–78.

[112] S. Kiyomoto, M. S. Rahman, and A. Basu, "On blockchain-based anonymized dataset distribution platform," in *2017 IEEE 15th International Conference on Software Engineering Research, Management and Applications (SERA).* IEEE, 2017, pp. 85–92.

[113] J. Lin, Z. Shen, A. Zhang, and Y. Chai, "Blockchain and iot based food traceability for smart agriculture," in *Proceedings of the 3rd international conference on crowd science and engineering*, 2018, pp. 1–6.

[114] L. Wang, Y. He, and Z. Wu, "Design of a blockchain-enabled traceability system framework for food supply chains," *Foods*, vol. 11, no. 5, p. 744, 2022.

[115] F. Tian, "An agri-food supply chain traceability system for china based on rfid & blockchain technology," in *2016 13th international conference on service systems and service management (ICSSSM).* IEEE, 2016, pp. 1–6.

[116] T. K. Agrawal, V. Kumar, R. Pal, L. Wang, and Y. Chen, "Blockchain-based framework for supply chain traceability: A case example of textile and clothing industry," *Computers & industrial engineering*, vol. 154, p. 107130, 2021.

[117] "Data Ownership Definition," 2022. [Online]. Available: https://ori.hhs.gov

[118] S. Moeniralam, "A blockchain based data production traceability system," 2018.

[119] H. Cui, Z. Chen, Y. Xi, H. Chen, and J. Hao, "Iot data management and lineage traceability: A blockchain-based solution," in *2019 IEEE/CIC International Conference on Communications Workshops in China (ICCC Workshops)*. IEEE, 2019, pp. 239–244.

[120] Z. Wang, Y. Tian, and J. Zhu, "Data sharing and tracing scheme based on blockchain," in *2018 8th international conference on logistics, Informatics and Service Sciences (LISS)*. IEEE, 2018, pp. 1–6.

[121] R. P. Pinto, B. M. Silva, and P. R. Inacio, "A system for the promotion of traceability and ownership of health data using blockchain," *IEEE Access*, vol. 10, pp. 92 760–92 773, 2022.

[122] "Chainbridge docs." [Online]. Available: https://chainbridge.chainsafe.io/

[123] J. Burdges, A. Cevallos, P. Czaban, R. Habermeier, S. Hosseini, F. Lama, H. K. Alper, X. Luo, F. Shirazi, A. Stewart *et al.*, "Overview of polkadot and its design considerations," *arXiv preprint arXiv:2005.13456*, 2020.

[124] J. Kwon and E. Buchman, "Cosmos whitepaper," *A Netw. Distrib. Ledgers*, p. 27, 2019.

[125] D. Tanana, "Avalanche blockchain protocol for distributed computing security," in *2019 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*. IEEE, 2019, pp. 1–3.

[126] "Cardano docs." [Online]. Available: https://docs.cardano.org/

[127] Algorand, "Blockchain interoperability: What is it and why does it matter," 2021. [Online]. Available: https://algorand.com/

[128] "Polygon whitepaper." [Online]. Available: https://polygon.technology/

[129] E. Abebe, D. Behl, C. Govindarajan, Y. Hu, D. Karunamoorthy, P. Novotny, V. Pandit, V. Ramakrishna, and C. Vecchiola, "Enabling Enterprise Blockchain Interoperability with Trusted Data Transfer (Industry Track)," in *Proceedings of the 20th International Middleware Conference Industrial Track*, 2019, pp. 29–35.

[130] Z. Ma, "Digital Rights Management: Model, Technology and Application," *China Communications*, vol. 14, no. 6, pp. 156–167, 2017.

[131] D. Bhowmik and T. Feng, "The Multimedia Blockchain: A Distributed and Tamper-proof Media Transaction Framework," in *2017 22nd International Conference on Digital Signal Processing (DSP)*. IEEE, 2017, pp. 1–5.

[132] A. Garba, A. D. Dwivedi, M. Kamal, G. Srivastava, M. Tariq, M. A. Hasan, and Z. Chen, “A Digital Rights Management System Based on a Scalable Blockchain,” *Peer-to-Peer Networking and Applications*, pp. 1–16, 2020.

[133] Z. Meng, T. Morizumi, S. Miyata, and H. Kinoshita, “Design Scheme of Copyright Management System Based on Digital Watermarking and Blockchain,” in *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*. IEEE, 2018, pp. 359–364.

[134] Z. Xu, L. Wei, J. Wu, and C. Long, “A Blockchain-Based Digital Copyright Protection System with Security and Efficiency,” in *CCF China Blockchain Conference*. Springer, 2020, pp. 34–49.

[135] Q. Wang, R. Li, Q. Wang, and S. Chen, “Non-fungible token (nft): Overview, evaluation, opportunities and challenges,” *arXiv preprint arXiv:2105.07447*, 2021.

[136] Z. Ma, M. Jiang, H. Gao, and Z. Wang, “Blockchain for Digital Rights Management,” *Future Generation Computer Systems*, vol. 89, pp. 746–764, 2018.

[137] C. Liu, X. Chen, J. Li, S. Yang, and Y. Sun, “A Novel Data Traceability Model Based on Blockchain and Digital Watermarking in Edge Computing,” in *Journal of Physics: Conference Series*, vol. 1682. IOP Publishing, 2020, p. 012041.

[138] B. Zhao, L. Fang, H. Zhang, C. Ge, W. Meng, L. Liu, and C. Su, “Y-DWMS: A Digital Watermark Management System Based on Smart Contracts,” *Sensors*, vol. 19, no. 14, p. 3091, 2019.

[139] S. Sahoo and R. Halder, “Traceability and Ownership Claim of Data on Big Data Marketplace using Blockchain Technology,” *Journal of Information and Telecommunication*, pp. 1–27, 2020.

[140] C.-H. V. Lin, C.-C. J. Huang, Y.-H. Yuan, and Z.-s. S. Yuan, “A Fully Decentralized Infrastructure for Subscription-based IoT Data Trading,” in *2020 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 2020, pp. 162–169.

[141] J. Gao, T. Wu, and X. Li, “Secure, Fair and Instant Data Trading Scheme Based on Bitcoin,” *Journal of Information Security and Applications*, vol. 53, p. 102511, 2020.

[142] W. Badreddine, K. Zhang, and C. Talhi, “Monetization using Blockchains for IoT Data Marketplace,” in *2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 2020, pp. 1–9.

[143] Y. Huang, Y. Zeng, F. Ye, and Y. Yang, "Fair and Protected Profit Sharing for Data Trading in Pervasive Edge Computing Environments," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications.* IEEE, 2020, pp. 1718–1727.

[144] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *International journal of web and grid services*, vol. 14, no. 4, pp. 352–375, 2018.

[145] S. Purdon, B. Kusy, R. Jurdak, and G. Challen, "Model-free hvac control using occupant feedback," in *38th Annual IEEE Conference on Local Computer Networks-Workshops.* IEEE, 2013, pp. 84–92.

[146] R. Xu, G. S. Ramachandran, Y. Chen, and B. Krishnamachari, "Blendsm-ddm: Blockchain-enabled secure microservices for decentralized data marketplaces," in *2019 IEEE international smart cities conference (ISC2).* IEEE, 2019, pp. 14–17.

[147] D. Niyato, D. T. Hoang, N. C. Luong, P. Wang, D. I. Kim, and Z. Han, "Smart data pricing models for the internet of things: a bundling strategy approach," *IEEE Network*, vol. 30, no. 2, pp. 18–25, 2016.

[148] D. Niyato, M. A. Alsheikh, P. Wang, D. I. Kim, and Z. Han, "Market model and optimal pricing scheme of big data and internet of things (iot)," in *2016 IEEE international conference on communications (ICC).* IEEE, 2016, pp. 1–6.

[149] W. Mao, Z. Zheng, and F. Wu, "Pricing for revenue maximization in iot data markets: An information design perspective," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications.* IEEE, 2019, pp. 1837–1845.

[150] D. Niyato, X. Lu, P. Wang, D. I. Kim, and Z. Han, "Economics of internet of things (iot): An information market approach," *Computer Science*, 2012.

[151] S. Malik, V. Dedeoglu, S. S. Kanhere, and R. Jurdak, "Trustchain: Trust management in blockchain and iot supported supply chains," in *2019 IEEE International Conference on Blockchain (Blockchain).* IEEE, 2019, pp. 184–193.

[152] C. Perera, S. Y. Wakenshaw, T. Baarslag, H. Haddadi, A. K. Bandara, R. Mortier, A. Crabtree, I. C. Ng, D. McAuley, and J. Crowcroft, "Valorising the iot databox: creating value for everyone," *Transactions on Emerging Telecommunications Technologies*, vol. 28, no. 1, p. e3125, 2017.

[153] Y. Shanmugarasa, H.-Y. Paik, S. S. Kanhere, and L. Zhu, "Towards automated data sharing in personal data stores," in *2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops).* IEEE, 2021, pp. 328–331.

[154] A. V. Sambra, E. Mansour, S. Hawke, M. Zereba, N. Greco, A. Ghanem, D. Zagidulin, A. Aboulnaga, and T. Berners-Lee, "Solid: a platform for decentralized social applications based on linked data," *MIT CSAIL & Qatar Computing Research Institute, Tech. Rep.*, 2016.

[155] J. Benet, "Ipfs-content addressed, versioned, p2p file system," *arXiv preprint arXiv:1407.3561*, 2014.

[156] B. Campbell, B. Ghena, Y.-S. Kuo, and P. Dutta, "Swarm gateway: Demo abstract," in *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*, 2016, pp. 217–218.

[157] Z. Liu, Y. Xiang, J. Shi, P. Gao, H. Wang, X. Xiao, B. Wen, Q. Li, and Y.-C. Hu, "Make web3.0 connected," *IEEE Transactions on Dependable and Secure Computing*, 2021.

[158] V. Charpenay, H. Nguyen, M. Ibrahim, A. Zappa, and A. Bröring, "Matching offerings and queries on an internet of things marketplace," in *European Semantic Web Conference.* Springer, 2018, pp. 66–71.

[159] H. R. Hasan and K. Salah, "Proof of delivery of digital assets using blockchain and smart contracts," *IEEE Access*, vol. 6, pp. 65 439–65 448, 2018.

[160] M. Dawande, J. Kalagnanam, P. Keskinocak, F. S. Salman, and R. Ravi, "Approximation algorithms for the multiple knapsack problem with assignment restrictions," *Journal of combinatorial optimization*, vol. 4, no. 2, pp. 171–186, 2000.

[161] G. Dahl and N. Foldnes, "Lp based heuristics for the multiple knapsack problem with assignment restrictions," *Annals of Operations Research*, vol. 146, no. 1, pp. 91–104, 2006.

[162] A. Murthy, C. Yeshwanth, and S. Rao, "Distributed approximation algorithms for the multiple knapsack problem," *arXiv preprint arXiv:1702.00787*, 2017.

[163] V. Molina, M. Kersten-Oertel, and T. Glatard, "A conceptual marketplace model for iot generated personal data," *arXiv preprint arXiv:1907.03047*, 2019.

[164] J. Byabazaire, G. O'Hare, and D. Delaney, "Data quality and trust: review of challenges and opportunities for data sharing in iot," *Electronics*, vol. 9, no. 12, p. 2083, 2020.

[165] S. Abbas, M. Merabti, and D. Llewellyn-Jones, “Deterring whitewashing attacks in reputation based schemes for mobile ad hoc networks,” in *2010 IFIP Wireless Days.* IEEE, 2010, pp. 1–6.

[166] G. Lax and G. M. Sarné, “Celltrust: a reputation model for c2c commerce,” *Electronic Commerce Research*, vol. 8, no. 4, pp. 193–216, 2008.

[167] “Personal data: The emergence of a new asset class,” 2011. [Online]. Available: https://www.weforum.org/reports/personal-data-emergence-new-asset-class/

[168] E. Onu, M. M. Kwakye, and K. Barker, “Contextual privacy policy modeling in iot,” in *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech).* IEEE, 2020, pp. 94–102.

[169] F. Schaub, R. Balebako, A. L. Durity, and L. F. Cranor, “A design space for effective privacy notices,” in *Eleventh symposium on usable privacy and security (SOUPS 2015)*, 2015, pp. 1–17.

[170] B. P. Knijnenburg, X. Page, P. Wisniewski, H. R. Lipford, N. Proferes, and J. Romano, “Modern socio-technical perspectives on privacy,” 2022.

[171] A. Abuarqoub and M. Hammoudeh, “Behaviour profiling in healthcare applications using the internet of things technology,” in *Proceedings of Fourth International Conference on Advances in Information Processing and Communication Technology.* Institute of Research Engineers and Doctors, 2016, pp. 1–4.

[172] J. Fernquist, T. Fängström, and L. Kaati, “Iot data profiles: The routines of your life reveals who you are,” in *2017 European Intelligence and Security Informatics Conference (EISIC).* IEEE, 2017, pp. 61–67.

[173] N. Helberger, “Profiling and targeting consumers in the internet of things–a new challenge for consumer law,” *Available at SSRN 2728717*, 2016.

[174] M. Hatamian and J. Serna-Olvera, “Beacon alarming: Informed decision-making supporter and privacy risk analyser in smartphone applications,” in *2017 IEEE International Conference on Consumer Electronics (ICCE).* IEEE, 2017, pp. 468–471.

[175] K. Hill, “How target figured out a teen girl was pregnant before her father did,” *Forbes, Inc*, vol. 7, pp. 4–1, 2012.

[176] M. Altman, A. B. Wood, D. O'Brien, and U. Gasser, "Practical approaches to big data privacy over time," *International Data Privacy Law*, 2018.

[177] J. Kröger, "Unexpected inferences from sensor data: a hidden privacy threat in the internet of things," in *IFIP International Internet of Things Conference*. Springer, 2018, pp. 147–159.

[178] S. Banerjea, S. Srivastava, and S. Kumar, "Data security in the internet of things: Challenges and opportunities," *Big Data Analytics for Internet of Things*, pp. 265–284, 2021.

[179] C. Ni, L. S. Cang, P. Gope, and G. Min, "Data anonymization evaluation for big data and iot environment," *Information Sciences*, vol. 605, pp. 381–392, 2022.

[180] Z. Tan, C. Wang, X. Fu, J. Cui, C. Jiang, and W. Han, "Re-identification of vehicular location-based metadata," *EAI Endorsed Transactions on Security and Safety*, vol. 4, no. 11, 2017.

[181] Y.-A. De Montjoye, E. Shmueli, S. S. Wang, and A. S. Pentland, "openpds: Protecting the privacy of metadata through safeanswers," *PloS one*, vol. 9, no. 7, p. e98790, 2014.

[182] D. I. Kaplun, D. V. Gnezdilov, G. A. Efimenko, A. M. Sinitca, and V. V. Gulvanskiy, "Research and implementation of the algorithm for data de-identification for internet of things," in *2017 IEEE II International Conference on Control in Technical Systems (CTS)*. IEEE, 2017, pp. 363–366.

[183] O. O. of the Information, P. Commissioner, A. Cavoukian, and D. Castro, *Big data and innovation, setting the record straight: de-identification does work.* Information and Privacy Commissioner, Ontario, 2014.

[184] A. Cavoukian and J. Jonas, *Privacy by design in the age of big data.* Information and Privacy Commissioner of Ontario, Canada, 2012.

[185] M. M. Madine, A. A. Battah, I. Yaqoob, K. Salah, R. Jayaraman, Y. Al-Hammadi, S. Pesic, and S. Ellahham, "Blockchain for giving patients control over their medical records," *IEEE Access*, vol. 8, pp. 193 102–193 115, 2020.

[186] A. Chaer, K. Salah, C. Lima, P. P. Ray, and T. Sheltami, "Blockchain for 5g: Opportunities and challenges," in *2019 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2019, pp. 1–6.

[187] Y. Zou, A. H. Mhaidli, A. McCall, and F. Schaub, "" i've got nothing to lose": Consumers' risk perceptions and protective actions after the equifax data breach." in *SOUPS@ USENIX Security Symposium*, 2018, pp. 197–216.

[188] S. Sahoo and R. Halder, “Traceability and ownership claim of data on big data marketplace using blockchain technology,” *Journal of Information and Telecommunication*, vol. 5, no. 1, pp. 35–61, 2021.

[189] L. Cheng, F. Liu, and D. Yao, “Enterprise data breach: causes, challenges, prevention, and future directions,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 7, no. 5, p. e1211, 2017.

[190] S. Alneyadi, E. Sithirasenan, and V. Muthukkumarasamy, “A survey on data leakage prevention systems,” *Journal of Network and Computer Applications*, vol. 62, pp. 137–152, 2016.

[191] S. Peneti and B. P. Rani, “Data leakage prevention system with time stamp,” in *2016 International Conference on Information Communication and Embedded Systems (ICICES)*. IEEE, 2016, pp. 1–4.

[192] J. Daubert, A. Wiesmaier, and P. Kikiras, “A view on privacy & trust in iot,” in *2015 IEEE International Conference on Communication Workshop (ICCW)*. IEEE, 2015, pp. 2665–2670.

[193] A. Acquisti and J. Grossklags, “Privacy and rationality in individual decision making,” *IEEE security & privacy*, vol. 3, no. 1, pp. 26–33, 2005.

[194] J. S. Montgomery, *A Privacy Risk Scoring Framework for Mobile*. Brigham Young University, 2014.

[195] G. Bal, “Designing privacy indicators for smartphone app markets: A new perspective on the nature of privacy risks of apps,” *Conference on Information Systems*, 2014.

[196] “Food labeling & nutrition,” 2022. [Online]. Available: https://www.fda.gov/food/food-labeling-nutrition

[197] “How to use the energyguide label to shop for home appliances,” 2021. [Online]. Available: https://consumer.ftc.gov/articles/how-use-energyguide-label-shop-home-appliances

[198] “Energy-efficient products,” 2022. [Online]. Available: https://ec.europa.eu/info/energy-climate-change-environment/standards-tools-and-labels/products-labelling-rules-and-requirements/energy-label-and-ecodesign/energy-efficient-products

[199] S. Perez, “Google play launches its own privacy ‘nutrition labels,’ following similar effort by apple,” April 2022. [Online]. Available: https://techcrunch.com/2022/04/26/google-play-launches-its-own-privacy-nutrition-labels-following-similar-effort-by-apple/

[200] P. Emami-Naeini, J. Dheenadhayalan, Y. Agarwal, and L. F. Cranor, "Which privacy and security attributes most impact consumers' risk perception and willingness to purchase iot devices?" in *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2021, pp. 519–536.

[201] N. Lindsey, "New iot security ratings a positive development for internet of things," Dec 2019. [Online]. Available: https://www.cpomagazine.com/cyber-security/new-iot-security-ratings-a-positive-development-for-internet-of-things/

[202] "Complete guide to gdpr compliance," 2022. [Online]. Available: https://gdpr.eu/

[203] "Privacy assessments," 2022. [Online]. Available: https://www.oaic.gov.au/privacy/privacy-assessments

[204] "California consumer privacy act (ccpa)," 2022. [Online]. Available: https://oag.ca.gov/privacy/ccpa

[205] A. Toy and D. C. Hay, "Privacy auditing standards," *Auditing: A Journal of Practice & Theory*, vol. 34, no. 3, pp. 181–199, 2015.

[206] "SOC for service organizations," 2022. [Online]. Available: https://us.aicpa.org

[207] "Data breach preparation and response," 2022. [Online]. Available: https://www.oaic.gov.au/

[208] S. Malik, V. Dedeoglu, S. S. Kanhere, and R. Jurdak, "Privchain: Provenance and privacy preservation in blockchain enabled supply chains," in *2022 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 2022, pp. 157–166.

[209] P. Papadimitriou and H. Garcia-Molina, "Data leakage detection," *IEEE Transactions on knowledge and data engineering*, vol. 23, no. 1, pp. 51–63, 2010.

[210] B. Densham, "Three cyber-security strategies to mitigate the impact of a data breach," *Network Security*, vol. 2015, no. 1, pp. 5–8, 2015.

[211] A. F. Westin, "Social and political dimensions of privacy," *Journal of social issues*, vol. 59, no. 2, pp. 431–453, 2003.

[212] K. Barker, M. Askari, M. Banerjee, K. Ghazinour, B. Mackas, M. Majedi, S. Pun, and A. Williams, "A data privacy taxonomy," in *British National Conference on Databases*. Springer, 2009, pp. 42–54.

[213] D. Bertram, "Likert scales," *Retrieved November*, vol. 2, no. 10, pp. 1–10, 2007.

[214] K. P. Enright, "Privacy audit checklist," 2022. [Online]. Available: https://cyber.harvard.edu/ecommerce/privacyaudit.html

[215] M. Park, H. Oh, and K. Lee, "Security risk measurement for information leakage in iot-based smart homes from a situational awareness perspective," *Sensors*, vol. 19, no. 9, p. 2148, 2019.

[216] M. R. Asghar, G. Dán, D. Miorandi, and I. Chlamtac, "Smart meter data privacy: A survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2820–2835, 2017.

[217] S. Hui, Z. Wang, X. Hou, X. Wang, H. Wang, Y. Li, and D. Jin, "Systematically quantifying iot privacy leakage in mobile networks," *IEEE Internet of Things Journal*, vol. 8, no. 9, pp. 7115–7125, 2020.

[218] A. S. Sendi, M. Jabbarifar, M. Shajari, and M. Dagenais, "Femra: Fuzzy expert model for risk assessment," in *2010 Fifth International Conference on Internet Monitoring and Protection.* IEEE, 2010, pp. 48–53.

[219] K. Shang and Z. Hossen, "Applying fuzzy logic to risk assessment and decision-making," *Casualty Actuarial Society, Canadian Institute of Actuaries, Society of Actuaries*, pp. 1–59, 2013.

[220] J. Li, Y. Bai, and N. Zaman, "A fuzzy modeling approach for risk-based access control in ehealth cloud," in *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications.* IEEE, 2013, pp. 17–23.

[221] "Understanding and assessing risk in personal data breaches," 2022. [Online]. Available: https://ico.org.uk/for-organisations/sme-web-hub/understanding-and-assessing-risk-in-personal-data-breaches/

[222] W. Asif, I. G. Ray, and M. Rajarajan, "An attack tree based risk evaluation approach for the internet of things," in *Proceedings of the 8th International Conference on the Internet of Things*, 2018, pp. 1–8.

[223] F. Khan, J. H. Kim, L. Mathiassen, and R. Moore, "Data breach management: An integrated risk model," *Information & Management*, vol. 58, no. 1, p. 103392, 2021.

[224] R. Baskerville, P. Spagnoletti, and J. Kim, "Incident-centered information security: Managing a strategic balance between prevention and response," *Information & management*, vol. 51, no. 1, pp. 138–151, 2014.

[225] "Hashed sharding," 2022. [Online]. Available: https://www.mongodb.com/

[226] "Data contract," 2022. [Online]. Available: https://research.csiro.au/blockchainpatterns/general-patterns/contract-structural-patterns/data-contract/

[227] "Upgrading Smart Contracts." [Online]. Available: https://docs.openzeppelin.com/learn/upgrading-smart-contracts

[228] "At&t cybersecurity insights report: Securing the edge," 2022. [Online]. Available: https://www.business.att.com/categories/cybersecurity-insights-report.html

[229] E. Heaslip, "Cloud dlp and regulatory compliance: 3 things you must know," 2021. [Online]. Available: https://nightfall.ai/cloud-dlp-and-regulatory-compliance-3-things-you-must-know

[230] L. Ponciano, P. Barbosa, F. Brasileiro, A. Brito, and N. Andrade, "Designing for pragmatists and fundamentalists: Privacy concerns and attitudes on the internet of things," in *Proceedings of the XVI Brazilian Symposium on Human Factors in Computing Systems*, 2017, pp. 1–10.

[231] P. Kumaraguru and L. F. Cranor, *Privacy indexes: a survey of Westin's studies.* Carnegie Mellon University, School of Computer Science, Institute for ..., 2005.

[232] J. M. Urban and C. J. Hoofnagle, "The privacy pragmatic as privacy vulnerable," in *Symposium on Usable Privacy and Security (SOUPS 2014) Workshop on Privacy Personas and Segmentation (PPS)*, 2014.

[233] A. Woodruff, V. Pihur, S. Consolvo, L. Brandimarte, and A. Acquisti, "Would a privacy fundamentalist sell their dna for $1000... if nothing bad happened as a result? the westin categories, behavioral intentions, and consequences," in *10th Symposium On Usable Privacy and Security (SOUPS 2014)*, 2014, pp. 1–18.

[234] "Hyperledger Caliper," Apr 2020. [Online]. Available: https://www.hyperledger.org/use/caliper

[235] "Solidity," 2022. [Online]. Available: https://docs.soliditylang.org/en/v0.8.9/

[236] W. Wang, D. Niyato, P. Wang, and A. Leshem, "Decentralized caching for content delivery based on blockchain: A game theoretic perspective," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–6.

[237] J. Guo, C. Li, and Y. Luo, "Blockchain-assisted caching optimization and data storage methods in edge environment," *The Journal of Supercomputing*, pp. 1–33, 2022.

[238] F. Xiong, M. Xie, L. Zhao, C. Li, and X. Fan, “Recognition and evaluation of data as intangible assets,” *SAGE Open*, vol. 12, no. 2, p. 21582440221094600, 2022.

[239] P. Hummel, M. Braun, and P. Dabrock, “Own data? Ethical Reflections on Data Ownership,” *Philosophy & Technology*, pp. 1–28, 2020.

[240] I. Stepanov, “Introducing a Property Right over Data in the EU: the Data Producer’s Right–an Evaluation,” *International Review of Law, Computers & Technology*, vol. 34, no. 1, pp. 65–86, 2020.

[241] A. Yang, “Data as a Property Right - Yang2020 - Andrew Yang for President,” accessed: 2022-12-01. [Online]. Available: https://2020.yang2020.com/policies/data-property-right

[242] “About ArcGIS: Mapping & Analytics Software and Services.” [Online]. Available: https://www.esri.com/en-us/arcgis/about-arcgis/overview

[243] I. J. Cox, M. L. Miller, and J. A. Bloom, “Watermarking Applications and their Properties,” in *Proceedings International Conference on Information Technology: Coding and Computing (Cat. No. PR00540).* IEEE, 2000, pp. 6–10.

[244] S. Katzenbeisser and F. Petitcolas, “Digital watermarking,” *Artech House, London*, vol. 2, 2000.

[245] R. Xiao, X. Sun, and Y. Yang, “Copyright Protection in Wireless Sensor Networks by Watermarking,” in *2008 International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, 2008, pp. 7–10.

[246] B. Harjito, V. Potdar, and J. Singh, “Watermarking Technique for Copyright Protection of Wireless Sensor Network Data using LFSR and Kolmogorov Complexity,” in *Proceedings of the 10th International Conference on Advances in Mobile Computing & Multimedia*, 2012, pp. 208–217.

[247] C. Luo, A. Fylakis, J. Partala, S. Klakegg, J. Goncalves, K. Liang, T. Seppänen, and V. Kostakos, “A Data Hiding Approach for Sensitive Smartphone Data,” in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2016, pp. 557–568.

[248] H. Guo, Y. Li, and S. Jajodia, “Chaining Watermarks for Detecting Malicious Modifications to Streaming Data,” *Information Sciences*, vol. 177, no. 1, pp. 281–298, 2007.

[249] P. Gupta, V. Dedeoglu, K. Najeebullah, S. S. Kanhere, and R. Jurdak, “Energy-aware Demand Selection and Allocation for Real-time IoT Data Trading,” in *2020 IEEE International Conference on Smart Computing (SMARTCOMP).* IEEE, 2020, pp. 138–147.

[250] “Erc 20.” [Online]. Available: https://docs.openzeppelin.com/contracts/2.x/api/token/erc20

[251] I. Kamel and H. Juma, “A Lightweight Data Integrity Scheme for Sensor Networks,” *Sensors*, vol. 11, no. 4, pp. 4118–4136, 2011.

[252] S. King and S. Nadal, “Ppcoin: Peer-to-peer Crypto-currency with Proof-of-Stake,” *Self-published Paper, August*, vol. 19, no. 1, 2012.

[253] A. Manzoor, M. Liyanage, A. Braeke, S. S. Kanhere, and M. Ylianttila, “Blockchain Based Proxy Re-encryption Scheme for Secure IoT Data Sharing,” in *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 2019, pp. 99–103.

[254] C. Böhm and M. Hofer, *Physical Unclonable Functions in Theory and Practice.* Springer Science & Business Media, 2012.

[255] H. R. Hasan and K. Salah, “Proof of Delivery of Digital Assets Using Blockchain and Smart Contracts,” *IEEE Access*, vol. 6, pp. 65 439–65 448, 2018.

[256] Y. Jiang, C. Wang, Y. Wang, and L. Gao, “A Cross-Chain Solution to Integrating Multiple Blockchains for IoT Data Management,” *Sensors*, vol. 19, no. 9, p. 2042, 2019.

[257] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, “Blockchain for IoT Security and Privacy: The Case Study of a Smart Home,” in *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops).* IEEE, 2017, pp. 618–623.

[258] “Hash Time Locked Contracts.” [Online]. Available: https://en.bitcoin.it/wiki/Hash_Time_Locked_Contracts

[259] M. Herlihy, “Atomic Cross-Chain Swaps,” in *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing*, 2018, pp. 245–254.

[260] H. Chai, S. Yang, Z. L. Jiang, and X. Wang, “A Robust and Reversible Watermarking Technique for Relational Dataset Based on Clustering,” in *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE).* IEEE, 2019, pp. 411–418.

[261] J. Xuehua, “Digital Watermarking and its Application in Image Copyright Protection,” in *2010 International Conference on Intelligent Computation Technology and Automation*, vol. 2. IEEE, 2010, pp. 114–117.

[262] A. Jadhav and M. Kolhekar, "Digital Watermarking in Video for Copyright Protection," in *2014 International Conference on Electronic Systems, Signal Processing and Computing Technologies*. IEEE, 2014, pp. 140–144.

[263] P. W. Eklund and R. Beck, "Factors that Impact Blockchain Scalability," in *Proceedings of the 11th International Conference on Management of Digital Ecosystems*, 2019, pp. 126–133.

[264] M. Dabbagh, K.-K. R. Choo, A. Beheshti, M. Tahir, and N. S. Safa, "A Survey of Empirical Performance Evaluation of Permissioned Blockchain Platforms: Challenges and Opportunities," *Computers & Security*, vol. 100, p. 102078, 2021.

[265] "Data Protection and Privacy Legislation Worldwide," 2022. [Online]. Available: https://unctad.org/page/data-protection-and-privacy-legislation-worldwide

[266] M. J. Amiri, D. Agrawal, and A. El Abbadi, "Sharper: Sharding permissioned blockchains over network clusters," in *Proceedings of the 2021 International Conference on Management of Data*, 2021, pp. 76–88.