sensors

MDPI

Article

DataMesh+: A Blockchain-Powered Peer-to-Peer Data Exchange Model for Self-Sovereign Data Marketplaces

Mpyana Mwamba Merlec [1] and Hoh Peter In [1,2,*]

[1] Department of Computer Science and Engineering, Korea University, 145 Anam-ro, Seongbuk-gu, Seoul 02841, Republic of Korea; mlecjm@korea.ac.kr
[2] DAO Solution, Inc., 169, Yeoksam-ro, Gangnam-gu, Seoul 06247, Republic of Korea
* Correspondence: hoh_in@korea.ac.kr (H.P.I.)

Abstract: In contemporary data-driven economies, data has become a valuable digital asset that is eligible for trading and monetization. Peer-to-peer (P2P) marketplaces play a crucial role in establishing direct connections between data providers and consumers. However, traditional data marketplaces exhibit inadequacies. Functioning as centralized platforms, they suffer from issues such as insufficient trust, transparency, fairness, accountability, and security. Moreover, users lack consent and ownership control over their data. To address these issues, we propose *DataMesh+*, an innovative blockchain-powered, decentralized P2P data exchange model for self-sovereign data marketplaces. This user-centric decentralized approach leverages blockchain-based smart contracts to enable fair, transparent, reliable, and secure data trading marketplaces, empowering users to retain full sovereignty and control over their data. In this article, we describe the design and implementation of our approach, which was developed to demonstrate its feasibility. We evaluated the model's acceptability and reliability through experimental testing and validation. Furthermore, we assessed the security and performance in terms of smart contract deployment and transaction execution costs, as well as the blockchain and storage network performance.

Keywords: blockchain; data marketplace; data mesh; peer-to-peer data trading; self-sovereign data marketplace (SSDM); smart contracts

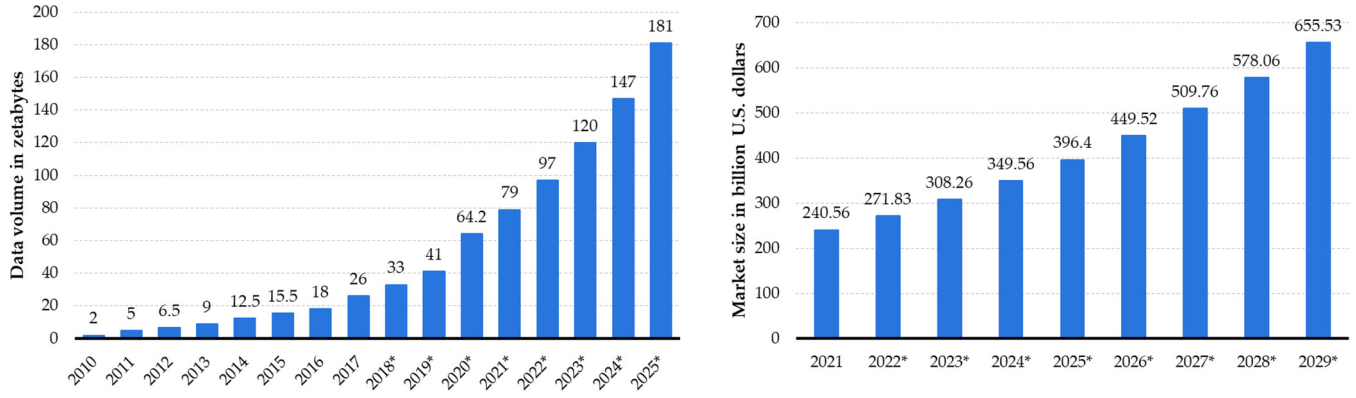
Citation: Merlec, M.M.; In, H.P. DataMesh+: A Blockchain-Powered Peer-to-Peer Data Exchange Model for Self-Sovereign Data Marketplaces. *Sensors* **2024**, *24*, 1896. https://doi.org/10.3390/s24061896

Academic Editor: Paolo Trunfio

Received: 13 January 2024
Revised: 19 February 2024
Accepted: 8 March 2024
Published: 15 March 2024

1. Introduction

In the evolving landscape of data-driven economies of scale, data emerges as a valuable digital asset ripe for trading and monetization. According to [1,2], projections indicate a surge in the global data sphere to 181 ZB by 2025, as shown in Figure 1a, alongside an anticipated revenue boost for the global big data market to 655.53 billion dollars by 2029, as shown in Figure 1b. The convergence of mobile cloud computing and communications, the Internet of things (IoT), artificial intelligence (AI), big data analytics, and blockchain technologies has created unprecedented economic prospects for individuals and organizations to capitalize on their data [3–18]. However, the path to effective monetization of data is riddled with challenges, mainly stemming from the limitations of traditional online data marketplaces [4–12]. To address these challenges, the establishment of peer-to-peer (P2P) marketplaces is imperative, facilitating direct transactions between data providers (sellers) and consumers (buyers) over the Internet [7–18]. A P2P data marketplace is an internet-based marketplace, also referred to as an electronic marketplace (e-marketplace) platform where users can connect to directly exchange, sell or buy data with or without the involvement of intermediaries [12–14]. In the inherently trust-challenged realm of the Internet, trusted third parties (TTPs) play a vital role in fostering trust and resolving disputes among transacting parties [15–17]. Powered by a community comprising data providers and consumers, P2P marketplaces require a robust infrastructure to enable secure, fair, transparent, and reliable transactions, along with seamless payment processing [10–22].



(**a**) (**b**)

Figure 1. (**a**) Amount of data created, collected, and used worldwide from 2010 with predictions up to 2025 (in zettabytes) [1]; (**b**) Global big data analytics market size from 2021 to 2029 (in billions of US dollars) [2].

However, traditional data marketplaces are inadequate. Operating as centralized platforms, they lack the necessary levels of trust, transparency, fairness, accountability, and security [3–22]. Despite their extensive adoption, these centralized data marketplaces are often face vulnerabilities such as single points of failure (SPF) and fail to offer adequate ownership and consent control over data use [16–20], thus contravening data protection regulations such as the General Data Protection Regulation (GDPR) [23]. Furthermore, their reliance on TTPs leads to increased transaction costs, inequitable revenue distribution, and increased complexity [11,17,22]. Consequently, users remain uncertain about how their data are collected and used, leading to mistrust and insufficiency in data transactions [18–21].

The emergence of *data mesh* architecture [24,25], *blockchain* [26–28], and *decentralized P2P storage* technologies [29] offers a paradigmatic solution to these pressing problems. *Data mesh* is a new conceptual data architecture framework that emphasizes decentralized ownership and control, treats data as a product, and focuses on domain-driven design [24,25]. This approach enhances value extraction from data by overcoming the limitations of traditional centralized data systems across various business domains within or across large and complex organizations [25,30]. Moreover, data mesh advances the establishment of a self-serving data infrastructure that enables domain teams to access and process data autonomously [30,31]. The core properties of blockchain–decentralization, immutability, and tamper resistance–herald a new era of secure, reliable, and transparent transactions [26,27]. By eliminating the need for central authorities, blockchain significantly reduces transaction costs while increasing efficiency [28,32]. The implementation of smart contracts on blockchain networks enables automated, trustless transactions characterized by transparency, auditability, and immutability [21,28,32]. This transition to a blockchain-based framework marks a stride toward decentralized P2P data marketplaces, offering a more equitable, secure, reliable, and transparent environment for data exchange. Despite advances in blockchain and smart contract-based P2P data trading models [12–19,21,22,33–51], existing systems have yet to fully address the inherent challenges of traditional marketplaces, particularly in terms of decentralization, transparency, fairness, security, trust, user control over data ownership, and consent control over data. How can blockchain, smart contracts, and decentralized storage technologies be leveraged to build secure, reliable, and user-centric decentralized data marketplaces?

To address these challenges, this study made the following contributions:

- We introduced *DataMesh+*, an innovative blockchain-powered P2P data exchange model for decentralized self-sovereign data marketplaces (SSDMs). DataMesh+ advances the data mesh concept by integrating blockchain and decentralized storage technologies to enhance decentralization in data trading. It prioritizes user control by

employing blockchain-based smart contracts to enable fair, transparent, reliable, and secure data trading marketplaces and empowers users to be sovereign and retain full control over their data. Smart contracts execute self-enforcing agreements between buyers and sellers, facilitating trustworthy transactions among globally disparate and anonymous parties without relying on centralized TTPs.

- We leveraged the Ethereum blockchain [27] to build a prototype that validates the practicality and effectiveness of our approach. To achieve pseudo-anonymity, users are identified through *externally owned accounts* (*EOAs*) provided by the Ethereum blockchain, which are secured with private and public cryptographic key pairs. Data ownership is determined by public-private key pairs, digital signatures, and account addresses. Digital signatures authenticate participant identities by cryptographically verifying transaction origins and holding them accountable by providing verifiable proof of their involvement in blockchain-recorded activities. Smart contracts track the participant activities and enforce the consequences of their actions. The *InterPlanetary File System* (IPFS) [29] provides resilient and highly available data storage and sharing capabilities in a secure, decentralized, and censorship-resistant manner, thus increasing the robustness of the proposed framework.
- We provided a comprehensive literature review addressing the design considerations, principles, and challenges associated with the development of a decentralized P2P data marketplace based on blockchain.
- We have outlined the operational workflow and architectural framework of the proposed model and its implementation. We assessed the acceptability and reliability of the proposed model through experimental testing and validation. Furthermore, we evaluated the security and performance in terms of smart contract deployment and transaction execution costs, the blockchain, and storage network performance.

The subsequent sections of this article are structured as follows: Section 2 provides an in-depth exploration of the literature review and preliminary concepts; Section 3 describes the design and architecture of the proposed model; Section 4 discusses the model implementation and evaluation; Section 5 addresses the limitations and remaining challenges; and Section 6 concludes the paper and offers insights into potential future research directions.

2. Literature Review and Preliminaries

This section provides an overview of existing data marketplaces and trends towards blockchain- and smart contract-based P2P data trading approaches. Subsequently, it presents preliminary research concepts including blockchain technology, smart contracts, IPFS, access control mechanisms, cryptographic primitives, and digital signatures.

2.1. Online Data Marketplaces

Recent years have seen extensive research into online data marketplaces [3–19]. These platforms vary across multiple dimensions, including target industry, business model, underlying technology, system architecture, type of data and services offered, trading mechanisms, use cases, and functionalities [5–16]. Foundational studies by [3–12] established a conceptual foundation and explored the concept of data as a commodity in a digital economy. In addition, refs. [5,10,11] present comprehensive definition and classification frameworks for data marketplaces that consider certain characteristics, including value propositions, market positioning, market access, and control, and data governance [16–18]. Furthermore, they address system architecture, integration, data acquisition, matching mechanisms, transformation, pricing, and revenue models [5,11,17]. The exploration of real-time data dynamics within these marketplaces and their impact on privacy and transaction efficiency is documented in [6,7]. The challenges and mechanisms of trading personal data, especially in IoT data marketplaces, are discussed in [8]. Hatamian [9] identified the technological barriers that hinder the development of IoT data marketplaces. Contributions from studies in [10,11] have enriched the understanding of data marketplace business models and introduced a taxonomy that explains the economic fundamentals of

data trading. In [12], an architectural perspective for P2P data monetization in the context of fog computing is presented.

However, traditional data marketplaces, characterized by their centralized architecture, fail to deliver the required level of trust, transparency, fairness, accountability, and security [5–19,21,22]. Moreover, users face challenges regarding consent and ownership control over their data to protect privacy [7,14–20], as mandated by data protection regulations, such as GDPR [23]. Reliance on TTPs in existing centralized marketplaces increases transaction costs and creates friction in revenue sharing [8,14–22]. Due to insufficient transparency, fairness, and user consent control in centralized marketplaces, users remain unaware of the collection, extent, timing, and recipients of their data sales [20–22]. Moreover, centralized architectures encounter SPF problems because administrators have unrestricted power and authority over user data. Consequently, these systems become prime targets for attackers seeking economic gains.

2.2. Data Mesh, Blockchain and Smart Contracts

Data mesh architecture [24,25], blockchain technology [26–28], and decentralized P2P storage networks [29] offer diverse features that P2P SSDMs can use to address the problems and limitations of existing centralized systems. Data mesh represents a novel approach to data architecture, transitioning from traditional centralized data management systems to a decentralized and domain-centric model [24,25]. This concept, pioneered by Zhamak Dehghani [24], revolves around treating data as a product and is based on four main pillars: (1) domain-oriented decentralized data ownership, (2) data as a product, (3) self-managed data infrastructure, and (4) federated computing management. The objective is to decentralize and control data ownership by distributing it among various business units within or between organizations. Each division regards its data as a product that promotes better alignment with business needs and enhances flexibility and scalability [30,31]. Although this approach enhances organizational agility and democratizes data access, it remains independent of the underlying technologies for data storage and transactions.

The concept of blockchain originally emerged as the technology supporting the Bitcoin system [26], a P2P electronic payment system that uses a public, shared, and immutable ledger to record a continuously expanding list of transactions. A blockchain represents a type of distributed ledger technology (DLT) for recording a series of time-stamped transactions that are sequentially linked in blocks and cryptographically secured [26–28]. Altering transactions in a block requires retroactive updating of all preceding blocks and obtaining consensus from most network participants [26,27]. Ethereum, an open-source, decentralized blockchain platform, is distinguished by its incorporation of featuring smart contracts [27,32], which are computer programs running on the blockchain that are capable of automatic execution upon meeting predefined conditions. Smart contracts facilitate the automation, execution, control, or documentation of events and actions in accordance with contract terms [20,27,32]. They are essential for developing decentralized applications (DApps) that operate on the blockchain. In contrast to traditional applications, blockchain offers DApps enhanced security and transparency in transactions [28,32]. The use of smart contracts can improve efficiency, trust, and security across various of applications [20–22,26,32,33], including cross-border data trading and electronic payments. Ethereum incorporates a universal, Turing-complete, virtual state machine known as the *Ethereum Virtual Machine* (EVM) [27], specifically for the execution of smart contract code. Within an Ethereum network, two primary types of nodes exist: miner nodes and regular nodes. Regular nodes facilitate transaction forwarding within the network, while miners verify and validate transactions by mining new blocks [27,28]. To ensure the consistency of the blockchain, a consensus protocol synchronizes the state of the ledger for each node in the network. Ethereum supports multiple consensus protocols including *Proof-of-Work* (PoW), *Proof-of-Stake* (PoS), and *Proof-of-Authority* (PoA) [26–28].

Blockchain's key features include decentralization, transparency, tamper-resistance, and immutability. These attributes are facilitated by publicly accessible and verifiable distributed ledgers and storage that are consistent and secure in P2P networks [26–28].

Furthermore, blockchain technology offers transnational anonymity, immediacy, validity, traceability, and persistence [27,28]. In addition, business terms agreement can be embedded in smart contracts, which execute autonomously on the blockchain, free from censorship or third-party interference [22,27,32]. Thus, blockchain eliminates the need for a central authority to validate transactions, yielding significant computational and cost efficiencies [33–40]. Through smart contracts, novel trustless intermediation mechanisms for decentralized data marketplace services can emerge [41–45]. By eliminating outdated trust-building mechanisms in conventional e-marketplaces, marketplace intermediaries or TTPs are eliminated, thereby lowering barriers to entry and transaction costs [33–51]. The cryptographic security of blockchain and the decentralized nature of P2P storage reduce the risks associated with centralized data storage [20–22,27–29], such as data breaches and unauthorized access.

2.3. Blockchain-Enabled Data Marketplaces

Various initiatives are progressing toward blockchain- and smart contract-based P2P data trading approaches [13–22,33–51]. Table 1 outlines the comparison between the proposed approach and previous studies, considering the main features of the proposed model. Studies in [13–15,33–38] establish the foundational concepts for blockchain-based decentralized data trading platforms, facilitating direct transactions between buyers and sellers without the involvement of TTPs. Specifically, [13,14] advocate blockchain-based P2P marketplaces for data trading, emphasizing preservation of data ownership. The authors of [15] address trustless transactions in decentralized service marketplaces. This aligns with the trend toward decentralized marketplace systems, as shown in [16–18], which provides insights into the landscape of data marketplaces and their business models. The research endeavors of [17–19,21] shift the focus toward ensuring transaction integrity and the governance structures of blockchain-based data marketplaces. The role of smart contracts in establishing dynamic consent management systems is highlighted in [20], which emphasizes user control in digital data marketplaces. Wang et al. [22] demonstrated the application of blockchain in ensuring fair payment for cloud storage auditing, signifying the convergence of blockchain across various data marketplace applications. The studies of [33–35] explored the decentralized nature of blockchain-based marketplaces, whereas [36,37] proposed systems for the efficient trading of big data. In [34,38], the design considerations and implementation challenges of such trading platforms are explored, encompassing aspects such as fairness, efficiency, security, privacy, and regulatory compliance.

Table 1. Comparison of the proposed approach with previous works.

Ref./ Features	Data Mesh	P2P [1]	Blockchain	DDM [2]	SC [3]	IPFS [4]	W2WM [5]	COC [6]	DACS [7]	S & T [8]	T & T [9]	Fairness	Accountability	Built PE [10]	
[4]	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
[6]	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
[7]	✗	✗	✗	✗	✗	✗	✗	✓	✗	✓	✗	✗	✗	✓	✓
[8]	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
[12]	✗	✗	✓	✗	✓	✗	✗	✗	✗	✓	✓	✗	✓	✓	✗
[13]	✗	✓	✓	✗	✓	✗	✗	✗	✗	✓	✓	✗	✓	✓	✓
[14]	✗	✓	✓	✓	✓	✗	✗	✓	✗	✓	✓	✗	✓	✓	✗
[15]	✗	✓	✓	✓	✓	✓	✗	✗	✗	✓	✓	✗	✗	✓	✗
[34]	✗	✓	✓	✓	✓	✗	✗	✗	✗	✗	✓	✗	✓	✓	✓
[35]	✗	✓	✓	✗	✓	✗	✗	✗	✓	✓	✗	✗	✓	✗	✗
[37]	✗	✗	✓	✓	✓	✗	✗	✗	✓	✓	✓	✗	✓	✗	✗
[39]	✗	✓	✓	✓	✗	✗	✗	✓	✗	✓	✓	✗	✗	✓	✓
[40]	✗	✓	✓	✗	✓	✗	✗	✗	✗	✓	✓	✗	✓	✓	✓
[41]	✗	✗	✓	✓	✓	✓	✗	✗	✗	✓	✓	✗	✓	✓	✓
[42]	✗	✗	✓	✓	✓	✗	✗	✗	✗	✓	✓	✓	✓	✓	✓
[43]	✗	✗	✓	✓	✓	✗	✗	✗	✗	✓	✓	✗	✓	✓	✗
[45]	✗	✗	✓	✗	✓	✗	✗	✗	✗	✓	✓	✗	✗	✓	✗
[46]	✗	✗	✓	✓	✓	✗	✗	✗	✗	✓	✓	✗	✓	✗	✗
[47]	✗	✗	✓	✓	✓	✗	✗	✓	✓	✓	✗	✗	✓	✓	✗
[48]	✗	✗	✓	✓	✓	✗	✗	✗	✗	✓	✓	✗	✓	✓	✓

Table 1. *Cont.*

Ref./ Features	Data Mesh	P2P [1]	Block chain	DDM [2]	SC [3]	IPFS [4]	W2WM [5]	COC [6]	DACS [7]	S & T [8]	T & T [9]	Fairness	Account ability	Built PE [10]	
[49]	✗	✓	✓	✓	✓	✗	✗	✗	✗	✓	✓	✗	✓	✓	✓
[51]	✗	✓	✓	✓	✓	✗	✗	✓	✗	✓	✓	✗	✓	✓	✗
This work	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

[1] Peer-to-Peer, [2] Decentralized Data Marketplace, [3] Smart contract, [4] InterPlanetary File System, [5] Decentralized Access Control System, [6] Consent & Ownership Control [7] Wallet-to-Wallet massaging system for deal negotiation, [8] Security & Trust, [9] Transparency and Traceability, [10] Performance evaluation.

The decentralized marketplace model for digital content proposed by [39] further illustrates the application of blockchain in content distribution. Detailed descriptions of the essential requirements for a secure blockchain-based data trading ecosystem are provided in [40], while [41] explores the integration of machine learning with blockchain for data trading. In addition, [42] exemplifies a system developed for big data trading using blockchain, where smart contracts support data matching, price negotiation, and reward allocation. In contrast, [43] envisions a marketplace for self-managing machines with accurate and trustworthy data sources. The decentralized nature of blockchain as a basis for IoT data marketplaces is examined in [44–46]. The IDMob system [47] serves as a practical demonstration of a blockchain data marketplace, while [48] explores a trustless marketplace for IoT data. In addition, [49] investigated the security and efficiency of blockchain-based data trading systems for the Internet of vehicles (IoV). They introduced an iterative double data trading auction system for IoV, aiming to maximize social welfare using a consortium blockchain. In [50], the role of decentralized autonomous organizations (DAOs) in the production of citizen-generated data. Finally, the case study of the Wibson data marketplace by [51] provides insights into decentralized and privacy-preserving data trading.

Despite notable advances in research, achieving a comprehensive solution to the challenges of decentralization, transparency, fairness, security, trust, user consent, and data ownership in data marketplaces remains elusive. Our research aims to bridge these gaps by proposing DataMesh+, a robust model and architectural framework that complements the Data mesh model [24,25] and addresses these critical issues. This study offers a substantial contribution with a comprehensive and nuanced approach to designing data exchange for user-centric, secure, transparent, fair, and trustworthy decentralized P2P SSDMs empowered by blockchain technology and decentralized P2P storage networks.

2.4. Cryptographic Primitives and Digital Signature

Cryptography algorithms are indispensable for ensuring the authenticity, confidentiality, integrity, and non-repudiation of data. We have focused on asymmetric cryptography, specifically public key cryptography [52,53], as it plays a crucial role in modern information system security. Public key cryptography uses unique keys that are easy to compute but are based on mathematical functions whose inverses are difficult to compute. These cryptographic functions facilitate the creation of tamper-proof digital signatures and mathematically secured secrets. *Elliptic curve cryptography* (ECC) is a variant of asymmetric cryptography based on discrete logarithmic problems defined by adding and multiplying the points of an elliptic curve [53].

- *Elliptic curve digital signature algorithm* (ECDSA) [53] is used for signing and verifying transactions. Its key pairs are associated with certain domain parameters consisting of an elliptic curve E represented over a finite field ($\mathbb{F}_p$). E is characterized by a base point $G \in E(\mathbb{F}_p)$ [53]. In practice, the parameters of a domain D are defined as (q, FR, a, b, G, n, h), which include q, the field size, where q is equal to p, an odd prime number, or 2^m. FR denotes the field specification for elements of $\mathbb{F}_p$. Parameters a and b refer to the field elements of the elliptic curve E over $\mathbb{F}_p$, as defined in Equation (1) for the case where $p > 3$ and Equation (2) for the case where $p > 2$ [53].

$$E(\mathbb{F}_p) : y^2 + xy + y^3 = x^3 + ax + b \quad (1)$$

- G is a finite point of the curve, defined by x_G and y_G in $\mathbb{F}_p$ as $G = (x_G, y_G)$ with prime order in $E(\mathbb{F}_p)$. n is the order of G such that $n > 2160$ and $n > 4\sqrt{q}$ and h is the cofactor, defined as $h = \#E(\mathbb{F}_q)/n$ [53]. A domain can have parameters that are either shared by multiple entities or unique to a particular user.

$$E(\mathbb{F}_p) : y^2 = x^3 + ax + b \quad (2)$$

- *Secp256k1 curve*: Similar to Bitcoin, Ethereum uses secp256k1 [27,52], an elliptic curve primitive used for cryptographic encryption. Secp256k1 is defined over $\mathbb{F}_p$, where $p = 2^{256} - 2^{32} - 977$ with 256-bit prime order, as expressed in Equation (3) [52,53].

$$E(\mathbb{F}_q) : y^2 = x^3 + 7 \quad (3)$$

- *Private and public keys*: The private key is an arbitrary 256-bit integer k, multiplied by a predefined generator point G over the elliptic curve to produce another point from which the public key K is derived, as defined in Equation (4) [52,53]. The ECDSA key pair generation and validation algorithms are detailed in [53].

$$K = kG \quad (4)$$

- *Keccak256/SHA-3* (*Secure hash algorithm 3*): The Keccak256 hash function, also known as the *SHA-3* cryptographic hash algorithm [27], computes the hash value of data stored in a blockchain. Equation (5) computes the Keccak256 hash of a given memory input and returns a 32-byte size hash value.

$$Keccak256(bytes\ memory)\ returns(bytes32) \quad (5)$$

- *Ethereum address*: Ethereum addresses comprise 40 hexadecimal characters, unique identifiers derived from the corresponding ECDSA public key or the contract's Keccak256 hash function (specifically its last 20 bytes), as follows [27,52]. Note that when the address is computed, the prefix (hex) 04 of the public key is omitted.

$$A(k) = B_{96..25}\ (KEC(ECDSAPUBKEY(k))) \quad (6)$$

In the context of P2P data exchange, digital signatures play a pivotal role in ensuring the authenticity, integrity, and non-repudiation of data transactions between peers [29,52,53]. A digital signature is a cryptographic technique used to verify the authenticity and integrity of messages, documents, or transactions [52,53]. These signatures leverage mathematical algorithms to generate a unique digital fingerprint or signature for each piece of data. The signature is generated using the signatory's private key and can only be verified using the corresponding public key. In the Ethereum adaptation of ECDSA, transaction messages are signed by computing the Keccak256 hashes of the transaction input data encoded with recursive length prefixes (RLP) [52–54]. The result is a signature S, which is defined as follows [52]:

$$S = F_{sig}\ (F_{keccak256}\ (m), k), \quad (7)$$

where m represents the transaction serialized message, k denotes the EOA's signing private key, and F_{sig} and $F_{keccak256}$ correspond to the respective signing and Keccak256 hash functions, respectively. The signature creation algorithm is described in [52–54].

The verification is the reverse operation of a signature creation function for given signature values r and s, and the signer's public key K is used to compute a value X via Equation (8) [27,52]. X represents an elliptic curve point that serves as a temporary public key used for signature generation.

$$X \equiv (u_1 G + u_2 K) \quad (8)$$

The detailed specification of the signature verification algorithm can be found in [27,52]. It operates by using the message M, the signature's public key K, and the digital signature

S as inputs to compute point X. A valid signature is determined if x, the coordinate of the calculated point X, is equal to r. Digital signatures enhance the security and reliability of P2P data exchange by enabling secure authentication, preserving data integrity, and establishing non-repudiation mechanisms between participating peers.

2.5. InterPlanetary File System and Access Control

IPFS [29] is a P2P and distributed file storage and sharing system that leverages content-addressing to identify unique content files in a global namespace. This scheme enables off-chain storage of large files and embeds hyperlinks for each content within immutable transactions by time stamping and backing up the content without including the data in a blockchain [29,55]. Cryptographic hash algorithms paired with distributed multi-hash tables ensure data integrity. The essential properties of IPFS include decentralization, security, censorship resistance, high reliability and availability, and low network latency. However, due to content identifiers (CIDs) being globally accessible upon upload and pinning, IPFS inherently lacks confidentiality. Therefore, ensuring data confidentiality requires file encryption before the ciphertext is stored in the IPFS nodes instead of plain text. Decentralized access control and consent management empower users with data ownership and access control [20,55,56]. With the role-based access control (RBAC) scheme [57,58], user access rights and privileges can be assigned and managed based on their role in the system, such as a seller or buyer. Therefore, the integration of IPFS, blockchain, and smart contract-based decentralized consent and access control systems will fulfill the above-mentioned essential characteristics of the proposed P2P data marketplace model.

3. DataMesh+: Proposed Secure and Reliable Decentralized P2P Data Exchange Model

In this section, we discuss the intricate details of the proposed model, outline its operational workflow and architectural framework, and highlight key components that collectively form the innovative approach to secure, transparent, and efficient data trading.

3.1. DataMesh+ Model Overview

Figure 2 illustrates the operational overview of the proposed DataMesh+ model, which is a secure and reliable decentralized P2P data exchange model for SSDMs based on blockchain and decentralized storage technologies. This is a user-centric and decentralized approach that enables secure and reliable direct data trading transactions between peers without the need for a central authority or TTP. The DataMesh+ model leverages blockchain to ensure transaction integrity, transparency, traceability, and security. It uses smart contracts to automate fair trading processes and a decentralized P2P storage network to ensure that data remains secure and distributed, thus mitigating SPF concerns. In addition, it provides P2P communication channels (wallet-to-wallet, W2W) compatible with Ethereum addresses [59,60], enabling data sellers and buyers to negotiate and finalize their trades. In this context, the term S refers to a data owner who sells data (*Seller*), while B refers to a user who is willing to buy data (*Buyer*). The workflow of our proposed data trading approach depicted in Figure 2 unfolds as follows:

① First, data sellers, denoted as $S = \{S_1, S_2, \ldots, S_n\}$ create a set of data product profiles (DPP), defined as $DPP = \{DPP_1, DPP_2, \ldots, DPP_m\}$, which are indexed in the blockchain. These profiles are subsequently published on marketplaces to make them available for sale.

② Any data buyer B_i of a set $B = \{B_1, B_2, \ldots, B_k\}$ can search or query a specific DPP_{id} that matches her/his preferences.

③ The buyer sends a data product DPP_{id} order request, ③′ which is automatically forwarded to the corresponding data seller S_j.

Upon receiving the request, the seller confirms the order and sends an invoice. ④′ Simultaneously, a single-use decryption key and sample data are sent to the requester for verification and confirmation.

④ Upon receiving the sample data, the buyer verifies its correctness. The buyer can cancel the order request after deeming the data unsatisfactory.

⑤ If the buyer is satisfied with the sample data, he/she can make a payment, which is temporarily held in an escrow smart contract. The total amount paid (Tot_a) is the sum of data price P, service commission fee (C_φ), and transaction processing fee (T_φ), as defined in Equation (9). C_φ is the multiplication of data price P by α which is a predefined commission rate (i.e., α = 0.05%), as expressed by Equation (10). T_φ is the sum of the transaction-related costs, as defined in Equation (11). ⑥′ After receiving the payment notification, the seller verifies the payment status.

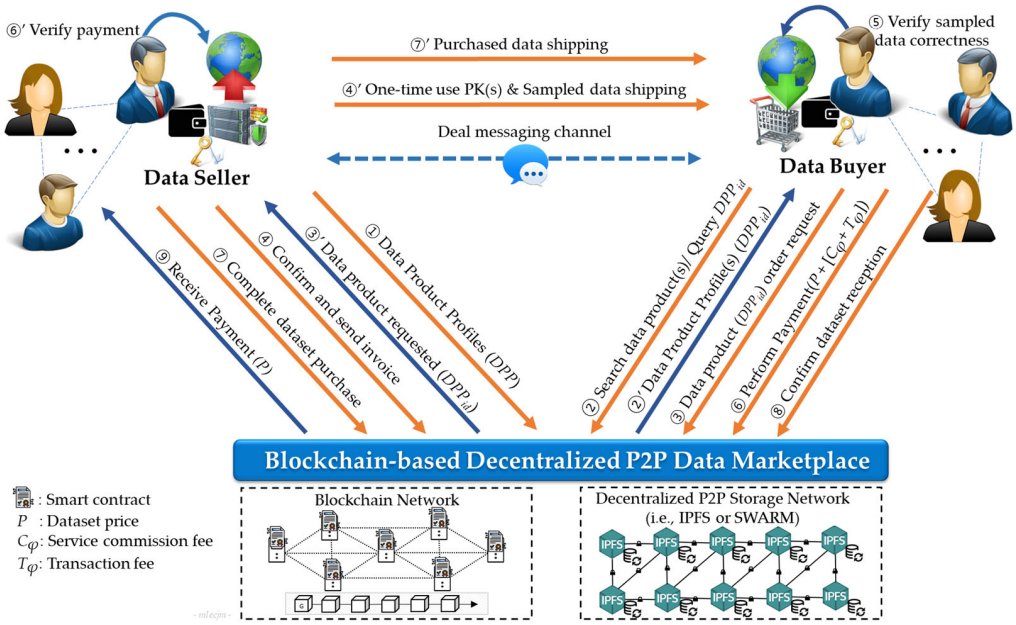


Figure 2. Overview of the proposed blockchain-powered P2P data exchange model (*DataMesh+*) for decentralized self-sovereign data marketplaces (SSDM).

$$Tot_a = P + C_\varphi + T_\varphi \tag{9}$$

⑥ Subsequently, the seller proceeds with the purchase and ships the purchased dataset to B_i, the corresponding buyer.

$$C_\varphi = \alpha\ P \tag{10}$$

⑦ Upon receiving the purchased dataset, the buyer verifies its correctness and ⑧′ confirms data reception for the seller to receive the payment. The buyer can cancel the deal and request a refund if the dataset is unsatisfactory.

⑧ Upon confirming data reception, the smart contract releases funds and transfers P into the seller's account.

$$T_\varphi = \sum_{i=1}^{n} \varepsilon_{\cos t}(T_i) \tag{11}$$

3.2. Architectural Framework

Figure 3 illustrates the multi-layered architectural framework of the proposed DataMesh+ model. The architecture is modular and segmented into five distinct layers, each serving specific functions within the marketplace platform.

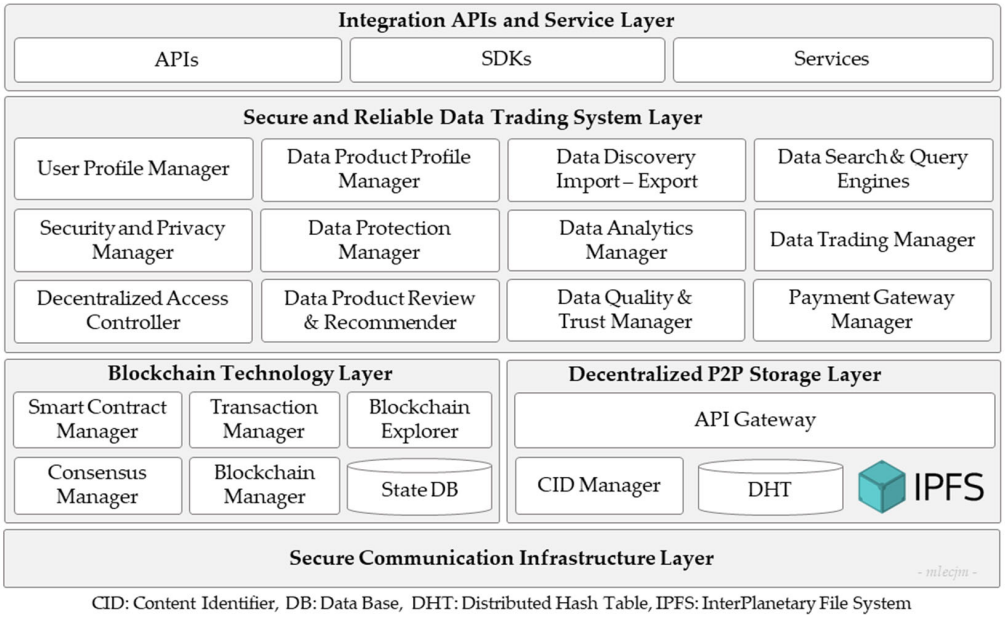


Figure 3. The architectural framework of the proposed blockchain-enabled P2P data exchange model for decentralized self-sovereign data marketplaces.

1. The **Integration APIs and Service Layer** provides the necessary application programming interfaces (APIs), software development kits (SDK), and services that facilitate interoperability, accessibility, and interaction between the data trading system layer and external systems or services. This layer enables various applications and services from data import/export to interaction with the system, thereby facilitating a diverse range of data exchange services and improving platform usability.
2. The **Secure and Reliable Data Trading System Layer** is a middleware that provides fair, transparent, secure, and trustworthy data trading features. It comprises the following principal modules: The *user profile manager* manages user profile credentials, roles, authentication, and authorization, ensuring accurate maintenance and secure access to user data. The *data product profile manager* allows the creation and management of data product profiles, which include metadata and usage terms associated with the datasets being traded. It also ensures accurate cataloging and retrievability of data products. The *data protection manager* enforces data protection policies, ensuring compliance with regulations and safeguarding data integrity. The *data search and query engines* facilitate efficient searching and querying within the marketplace, enabling users to find the data they need based on various search criteria. The *data discovery, import, and export* module facilitates the discovery, import, and export of data within the system, ensuring data integrity and accurate formatting during the process. The *data analytics manager* provides data processing and analytical capabilities, transforming raw data into actionable and valuable insights. The *data product review and recommender* collect user feedback on data products and recommend products to users based on their profiles, preferences, and past behavior. The *data quality and trust manager* maintains the data quality and manages trust scores for data products, ensuring adherence to high standards. The *data trading manager* facilitates data trading between users by overseeing transactions and enforcing the terms of agreed trade deals, including transaction validation, execution, and settlement. In addition, it provides features such as data product profile listing, price modeling, order management, and invoice management. The *payment gateway manager* handles financial transactions, allowing users to make payments for data products or receive payments within the marketplace. The *security and privacy manager* ensures that all transactions and data exchanges adhere to the highest security and privacy standards. The *decentralized*

access controller ensures that data access is governed by decentralized policies, thereby enhancing security and user control.

3. The **Blockchain Technology Layer** ensures the security, immutability, and transparency of transaction data. It facilitates the integration with the blockchain network for managing the distributed ledger and executing smart contracts. This layer comprises the following main components: The *smart contract manager* that govern the deployment, execution, and lifecycle of smart contracts governing data exchange agreements. The *transaction manager* acts as a facilitator within the blockchain, enabling the creation, validation, and processing of transactions. The *state DB* stores the current state of smart contracts and transaction data in a secure and accessible database. The *consensus manager* uses consensus algorithms to achieve agreement among network participants regarding the validity of transactions. The *blockchain manager* oversees the overall operation and maintenance of the blockchain network, including node management and network configuration. The *blockchain explorer* provides users with a graphical interface to explore and analyze blockchain data, including transaction history and smart contract details.
4. The **Decentralized P2P Storage Layer** oversees a reliable IPFS-based decentralized data storage and access control across a P2P storage network. This layer comprises the following components: The *API gateway* serves as an interface for accessing and interacting with the IPFS network, allowing users to upload, retrieve, and manage data stored on IPFS. The *CID manager* manages content identifiers (CIDs) for data stored on IPFS, ensuring unique identification and retrieval of stored content. The *distributed hash table (DHT*) facilitates decentralized peer discovery and routing within the IPFS network, enabling efficient data retrieval and distribution. The *IPFS protocol* enables storage and sharing of data in a distributed and censorship-resistant manner, leveraging a network of peer nodes to ensure data availability and integrity.
5. The **Secure Communication Infrastructure Layer** provides secure and reliable communication services built on dedicated secure Internet channels, such as scalability, control, and isolation on next-generation networks (SCION) [61]. SCION provides strong end-to-end encryption and protection against potential cyber threats [61]. Prioritizing scalability, control, and isolation, SCION enhances security compared with traditional Internet protocols. It uses path-aware routing and secure packet forwarding to mitigate common threats such as distributed denial-of-service (DDoS) attacks, route hijacking, and traffic analysis [61].

Together, these layers facilitate various functions within the system, leveraging blockchain technology for secure, transparent, and reliable market transactions. By using blockchain technology, the system ensures traceable and verifiable transactions, which promotes trust among users. In addition, P2P decentralized data storage solutions enhance the system's resilience and mitigate the risks associated with centralized data storage. This architectural framework is designed to establish a secure environment for data trading that ensures reliability, compliance, and user sovereignty in data management.

3.3. Data Trading Algorithms

The data are encrypted before being stored off-chain in a secure, censorship-resistant, highly available P2P decentralized storage network such as IPFS, as depicted in Figure 4. First, the plaintext file uploaded by the user is encrypted to generate its ciphertext, which is then signed and tagged with metadata containing the file name, extension, owner address, hash value, and access control list identifier. The signed file with the public metadata is then stored on the IPFS network and the corresponding CID and access URL are recorded in a blockchain-based immutable and shared ledger. Data access is governed by a decentralized RBAC system driven by smart contracts and supported by the blockchain to ensure robust security features and efficient rights management to protect the privacy and integrity of data in multi-party environments.

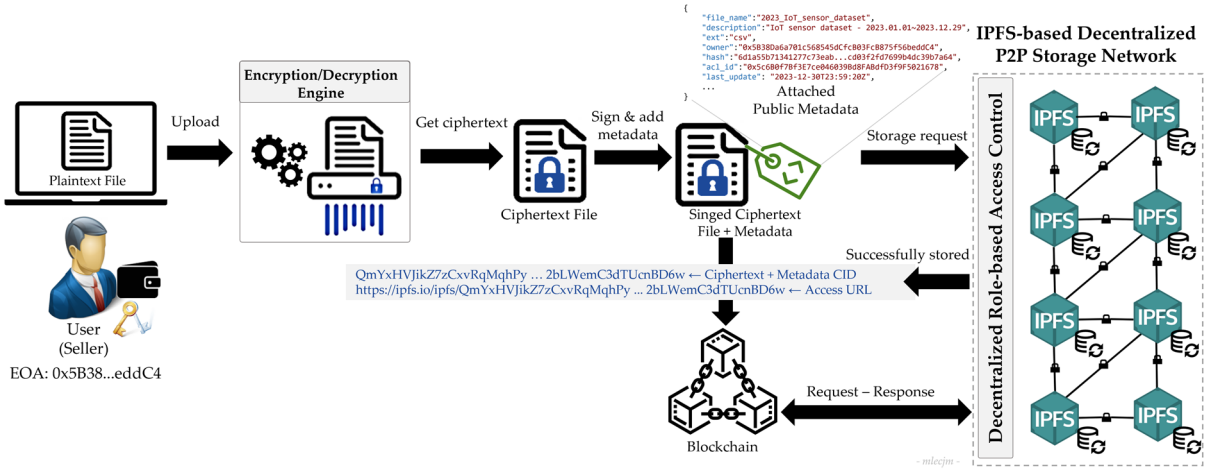


Figure 4. Data encryption before being stored in an IPFS-based decentralized P2P storage network.

Listing 1 outlines the metadata schema of the data product profile. Algorithm 1 outlines the procedure for creating and publishing the proposed data product profile (*DPP*). It uses input parameters such as the seller's identifier S_j and *DPP* details, including the data offering profile *id*, name η, type τ, owner's identifier ω, data hash h, digital signature s, previous owner's identifier υ, access URL ν, price p, currency c, and status σ. The algorithm first confirms whether the sender of the message is the owner of the data and then checks whether DPP_{id} does not exist in the blockchain to avoid duplicates. Subsequently, all input parameters are correctly assigned and transferred to the blockchain. After successful execution, the transaction hash and block number are returned.

Listing 1. Dataset profile data schema.

```
struct Dataset {
  string id
  string name
  string data_type;
  string description;
  string hash_value;
  uint256 size;
  string size_unit;
  uint256 price;
  string currency;
  string cid;
  string url;
  address owner;
  string signature;
  address previous_owner;
  bool isOpenForSell;
  bool isOrdered;
  bool isSold;
  uint256 lastUpdate;
}
```

Listing 2 shows the order data schema used by Algorithm 2, which represents the procedure for creating a data order. Algorithm 2 receives the order number O_{no}, DPP_{id}, and buyer identifier B_i as inputs. It first checks whether the buyer's identifier differs from the seller's and ensures that O_{no} does not already exist in the blockchain. Subsequently, all order parameters are saved, the states of the corresponding DPP_{id} are updated, and the order data record is transferred to the blockchain. Algorithm 3 describes the payment

procedure for an order that receives the order number O_{no}, buyer identifier B_i, and the payment amount P_a as inputs. It first checks whether B_i matches the buyer ID of the order O_{no}, then determines whether the balance exceeds (or at least matches) the total payment amount and whether the specified P_a matches this amount. Subsequently, it is determined whether the DPP_{id} status is "*Ordered*", "*Not Sold*", "*Not Paid*", and "*Not Cancelled*", before transferring the paid amount to the address of the escrow smart contract.

Listing 2. Purchase order data schema.

```
struct Order {
  string order
  string name
  address buyer;
  string dataset_id;
  uint256 price;
  string currency;
  uint256 fee_cost;
  uint256 total_amount;
  address seller;
  bool isPaid;
  bool isConfirmed;
  bool isCompleted;
  bool isCancalled;
  bool isRefundPaid;
  uint256 lastUpdate;
}
```

Algorithm 1: Data product profile creation

Parameters: Smart contract address SC_a, account address A_a
Input: Seller S_j and DPP details as $\{id, \eta, \tau, \omega, h, s, v, \nu, p, c, \sigma\}$
Output: T_{xh}, $Block_{no}$

if $S_j = \omega$ **then**
  Check whether DPP_{id} exists in the blockchain:
  $E \leftarrow sc.getDataset(DPP_{id})$
  if $E \neq$ NULL **then**
    Map each input parameter of DPP_{id}:
    $DPP[_{id}] \leftarrow \{\eta,\tau,\omega, h, s,v,\nu, p, c,\sigma\}$
    $DPP[_{id}].isOpenForSell \leftarrow true$
    $DPP[_{id}].isOrdered \leftarrow false$
    $DPP[_{id}].isSold \leftarrow false$
      $t = block.timestamp$
      $DPP[_{id}].lastUpdate \leftarrow t$
      Push $DPP[_{id}]$ instance to the blockchain
      Emit $NewDatasetAdded(DPP_{id}, \omega, \sigma, t)$ event
    else
      Return "Dataset profile DPP_{id} already exists."
    end if
  else
    Return "Only owner/seller can register dataset."
  end if
    Return the transaction execution state (T_{xh}, $Block_{no}$).

After a successful payment transfer, Algorithm 3 updates the states of DPP_{id} and O_{no} on the blockchain. Algorithm 4 handles the procedure for confirming receipt of the order, with O_{no} and B_i as input parameters. It first checks whether B_i is identical to the buyer's identifier from the order O_{no}. Subsequently, it checks whether the status of O_{no} is "*Paid*", "*Not Confirmed*", "*Not Completed*", and "*Not Cancelled*", and then updates the status of O_{no} as confirmed in the blockchain. Once the buyer confirms receipt of the order, the actual payment is transferred to the seller's account, and the data ownership record is updated according to Algorithm 5, which uses the message sender's address and the order identifier O_{no} as inputs. The algorithm checks whether the specified address is an authorized account of the smart contract administrator. Next, it confirms whether the escrow payment account holds sufficient funds for the payment amount to be transferred

P and whether the order status is "*Confirmed*", "*Not Completed*" and "*Not Cancelled*". If all conditions are satisfied, payment P is transferred to the seller's account S_j, followed by the transfer of DPP_{id} ownership on the blockchain. Subsequently, the related order and DPP states are updated accordingly. Finally, the block number and transaction hash are returned as proof that the transaction was completed successfully.

Algorithm 2: Data order creation

Parameters: Smart contract address SC_a, account address A_a
Input: O_{no}, DPP_{id}, B_i, T_φ
Output: T_{xh}, $Block_{no}$

if $B_i \neq DPP[_{id}].\omega$ **then**
  Check whether O_{no} exists in the blockchain:
  $E \leftarrow sc.getOrder(O_{no})$
  if $E \neq NULL$ **then**
    Map each input parameter of $O[_{no}]$:
    $O[_{no}]. \leftarrow \{DPP_{id}, B_i\}$
    $O[\text{no}].owner \leftarrow DPP[_{id}].\omega$
    $O[_{no}].price \leftarrow DPP[_{id}].P$
    $O[\text{no}].currency \leftarrow DPP[_{id}].c$
    $O[\text{no}].commissionFee \leftarrow DPP\ [_{id}].C_\varphi$
    $O[\text{no}].totalAmount \leftarrow (DPP[_{id}].P + DPP\ [_{id}].C_\varphi + T_\varphi)$
    $O[\text{no}].paymentAddr \leftarrow SC_a$
    $O[\text{no}].isPaid \leftarrow false$
    $O[\text{no}].isConfirmed \leftarrow false$
    $O[\text{no}].isCompleted \leftarrow false$
    $O[\text{no}].isRefunded \leftarrow false$
    $t = block.timestamp$
    Update DPP_{id} status:
    $DPP[_{id}].isOpenForSell \leftarrow false$
    $20.\ DPP[_{id}].isOrdered \leftarrow true$
    $DPP[_{id}].lastUpdate \leftarrow t$
    Push $O[_{no}]$ instance to the bsslockchain
      Emit $NewOrder(O_{no}, B_i, DPP_{id}, \omega, O_s, t)$ event
    else
      Return "Dataset order O_{no} already exists."
    else
      Return "Only buyers can create orders."
  Return the transaction execution state (T_{xh}, $Block_{no}$).

Algorithm 3: Order payment

Parameters: Smart contract address SC_a, account address A_a
Input: O_{no}, B_i, P_a // P_a: Paid amount
Output: T_{xh}, $Block_{no}$

if $B_i = O[_{no}].buyer$ **then**
  if $(B_i.balance \geq O[_{no}].T\ ot_a) \wedge (P_a = O[_{no}].T\ ot_a)$ **then**
    if $(DPP\ [_{id}].isOrdered = true) \wedge (DPP\ [_{id}].isSold = false)$ **then**
      if $(O[_{no}].isPaid = false) \wedge (O[_{no}].isCancelled = false)$ **then**
        Transfer $P_a$to A_a
        $O[_{no}].isPaid = true$
        $t = block.timestamp$
        Update DPP_{id} status:
        $DPP\ [_{id}].isSold \leftarrow true$
        $DPP\ [_{id}].lastUpdate \leftarrow t$
        Emit $OrderPaid(O_{no}, B_i, t)$ event
      else
      Return "$O[_{no}]$ status must be unpaid or not cancelled."
    else
      Return "$O[_{no}]$ status must be ordered and not sold."
    else
      Return "Insufficient balance or $P_a$is not equal to $T\ ot_a$"
  else
    Return "Different from order buyer account."
  Return the transaction execution state (T_{xh}, $Block_{no}$).

In the case of dataset order cancellation, Algorithm 6 executes by receiving the order identifier O_{no} and the buyer's account address B_i as inputs. It begins by validating whether the given account address matches the one included in the order, and then checks if the associated DPP's status is currently "*Ordered*", "*Not Completed*", and "*Not Cancelled*" yet. Subsequently, the DPP's status is updated as "*Cancelled*" on the blockchain. Upon successful order cancellation, the payment refund process is automatically initiated. Algorithm 7 highlights the process of refunding a payment to the buyer for a cancelled order. After

receiving a call with parameters, which includes the message sender address and order identifier O_{no}. Algorithm 7 verifies whether the message sender's address is an authorized smart contract's admin account. It then checks if the escrow payment account balance meets or exceeds the amount of payment P to be transferred and ensures that the order status is "*Paid*", "*Not Completed*", "*Not Cancelled*", and no refund has been executed. If these conditions are met, payment P is refunded to the respective buyer account B_i, and the related order and *DPP* states are updated on the blockchain accordingly. Finally, the transaction hash and block number are returned as a confirmation that the transaction was successfully executed. Figure 5 provides a summarized visualization of the proposed smart contract-based fair, secure, and reliable P2P data trading model for SSDMs.

Algorithm 4: Confirm order reception

Parameters: Smart contract address *SCa*, account address *Aa*
Input: O_{no}, B_i
Output: T_{xh}, $Block_{no}$

if $B_i = O[_{no}].buyer$ **then**
  if $(O[no].isPaid = true) \wedge (O[no].isConfirmed = false) \wedge (O[no].isCompleted = false) \wedge (O[no].isCancelled = false)$ **then**
    $O[no].isConfirmed \leftarrow true$
    Emit $OrderConfirmed(O_{no}, B_i, O_s, t)$ event
  else
  Return "Only B_i can confirm this order."
  Return the transaction execution state (T_{xh}, $Block_{no}$).

Algorithm 5: Payment transfer and dataset ownership update

Parameters: Smart contract address SC_a, account address A_a
Input: O_{no}, *msg.sender*
Output: T_{xh}, $Block_{no}$

if $msg.sender = A_a$ **then**
  if $SC_a.balance \geq P$ **then**
    if $(O[_{no}].isPaid = true) \wedge (O[_{no}].isConfirmed = true) \wedge (O[_{no}].isCompleted = false) \wedge (O[_{no}].isCancelled = false)$ **then**
      Transfer $O[_{no}].P$ to $O[_{no}].S_j$
      $prvOwner \leftarrow O[_{no}].DPP\ [_{id}].\omega$
      $t = block.timestamp$
      Update DPP_{id} status:
      $DPP\ [_{id}].owner \leftarrow O[_{no}].B_i$
      $DPP\ [_{id}].previousOwner \leftarrow prvOwner$
      $DPP\ [_{id}].isSold \leftarrow true$
      $DPP\ [_{id}].lastUpdate \leftarrow t$
        $O[_{no}].isCompleted \leftarrow true$
      Emit $PaymentSent(O_{no}, A_a, O_s, t)$ event
    else
      Return "$O[_{no}]$ status must be paid and confirmed."
  else
    Return "Insufficient balance."
else
    Return "Only admin can perform this operation."
Return the transaction execution state (T_{xh}, $Block_{no}$).

Algorithm 6: Dataset order cancellation

Parameters: Smart contract address SC_a, account address A_a
Input: O_{no}, B_i
Output: T_{xh}, $Block_{no}$

if $B_i = O[_{no}].buyer$ **then**
  if $(DPP\ [_{id}].isOrdered = true) \wedge (O[_{no}].isCompleted = false) \wedge (O[_{no}].isCancelled = false)$ **then**
    $O[_{no}]$.isCancelled ← true
    Emit $OrderCancelled(O_{no}, B_i, O_s, t)$ event
  else
    Return "$O[_{no}]$ status not be completed and cancelled."
else
    Return: "Only B_i can cancel the order."
Return the transaction execution state (T_{xh}, $Block_{no}$).

Algorithm 7: Payment refund to the buyer

Parameters: Smart contract address SC_a, account address A_a
Input: O_{no}, $msg.sender$
Output: T_{xh}, $Block_{no}$

if $msg.sender = A_a$ **then**
  if $SC_a.balance \geq P$ **then**
    if $(O[_{no}].isPaid = true) \wedge (O[_{no}].isCompleted = false) \wedge (O[_{no}].isCancelled = true) \wedge (O[_{no}].isRefunded = false)$ **then**
      Transfer $O[_{no}].P$ to $O[_{no}].B_i$
      Update DPP_{id} status: $DPP\ [_{id}].isOrdered \leftarrow false$
      $O[_{no}].isRefunded \leftarrow true$
      Emit $PaymentSent(O_{no}, A_a, O_s, t)$ event
  else
    Return: "Insufficient balance."
else
  Return: "Only admin can perform this operation."
Return the transaction execution state (T_{xh}, $Block_{no}$).

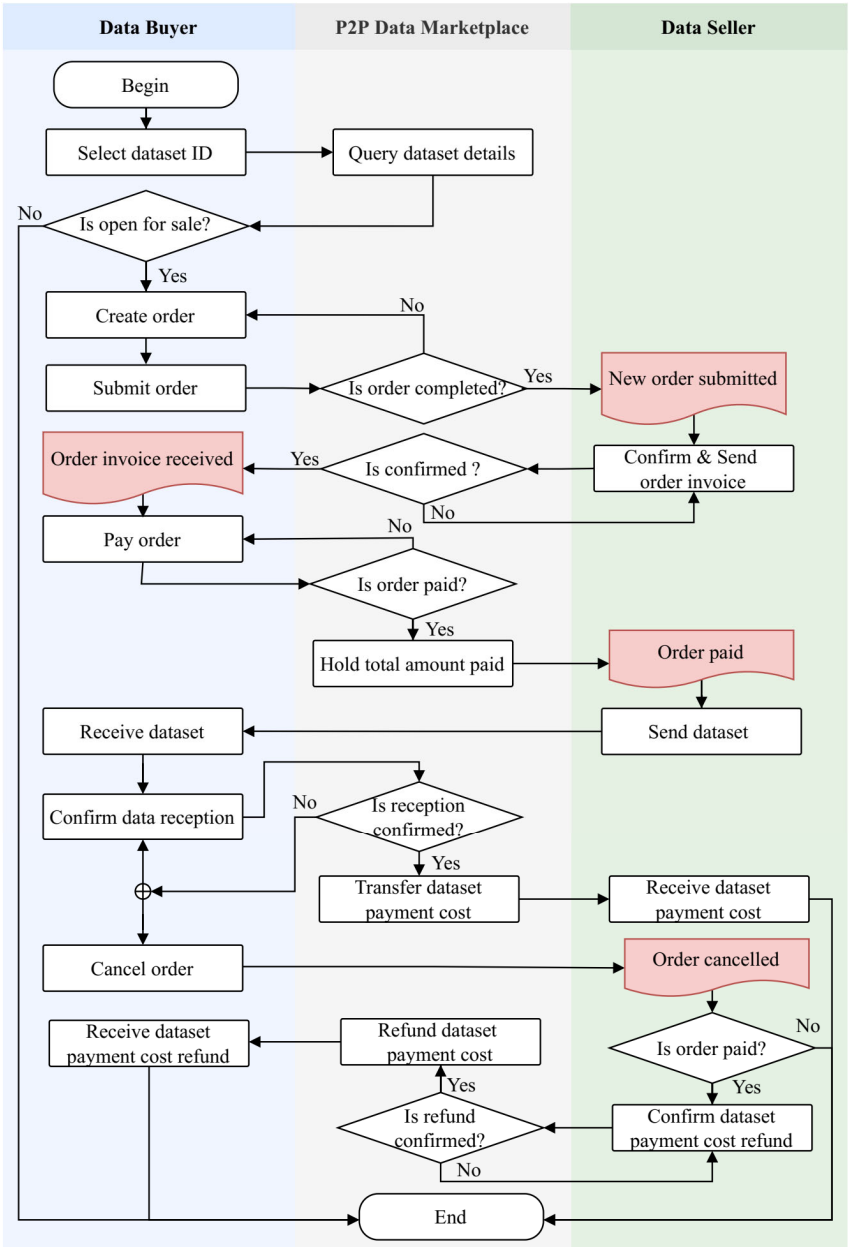


Figure 5. The smart contract-based fair, secure, and reliable P2P data trading flowchart of the proposed model for SSDMs.

4. Implementation and Evaluation

This section encompasses the considerations for implementing the proposed model and the subsequent experimental evaluation.

4.1. Experimental Environment Setup and Performance Metrics

Table 2 shows the experimental environment setup and key performance metrics. The smart contracts of the proposed system were developed in the Solidity programming language. They were then deployed and tested in the Goerli test network. Goerli is a cross-client test network for the Ethereum blockchain [27] that uses the PoA consensus protocol. The PoA algorithm is well suited for permissioned blockchains, where known validators from different organizations oversee network management. It provides higher performance with $O(n)$ computational complexity and can accommodate up to f faulty nodes within $2f + 1$ consensus nodes (n) [54]. For payment, we use the native Ethereum blockchain cryptocurrency, *Ether* (*ETH*) [27], which is created through the mining process as a reward for peer nodes that secure the network. *MetaMask* wallet is used for transaction signing and verification. The DApp is built using various programming languages, libraries, and frameworks including React, node.js, hardhat, and web3.js APIs. Remote procedure calls (RPCs) facilitate interaction with Ethereum nodes through smart contracts. Infura APIs provide secure, reliable, and scalable access to Ethereum and IPFS networks. To prevent network abuse issues and reward resource-providing nodes, Ethereum imposes fees for every programmable computation [27]. These fees, which are denominated in units of gas with equivalence in Ether, and cover various computations such as contract creation, account storage, state updates, and other execution of EVM operations [27,52].

Table 2. Experiment environment setup and performance metrics.

Parameters	Values
Network	Goerli Testnet (https://goerli.net/ accessed on 9 December 2022)
Network ID	5
Chain ID	5
Consensus Protocol	PoA Clique
Number of nodes	30
Epoch interval	30,000
Step period	15
Total difficulty	10,790,000
Gas limit	30,000,000
Average transaction size (bytes)	3760
Transaction throughput (TPS)	24
Average transaction latency (sec)	36
Average block size (KB)	102.34
Average number of Tx per block	60
Number of block confirmations	16
Average block time (sec)	15
Blockchain network utilization (%)	61.40
Smart contract language	Solidity
Compiler version	v0.8.17
EVM Version	London
Digital wallet	MetaMask v10.23.3
Dapp frameworks and IPAs	React, nodejs, hardhat, web3.js
Average encryption time per DPP (ms)	25.4
IPFS node & kubo agent	v0. 26.0 & v0.18.1
IPFS storage time per DPP object (ms)	39.6
Average size per DPP object (kb)	38
IPFS network bandwidth usage (kb/s)	109.70 (in)/83.12 (out)

4.2. Operational Cost Evaluation

To assess the feasibility and reliability of our model, we evaluated the deployment and operational costs of core smart contracts by considering the following parameters:

- *Smart contract deployment cost*: The deployment cost encompasses the code deposit, execution, and transaction costs. The code deposit cost is the maximum amount of gas

required for successful contract creation by placing the code into the state [27,52]. It varies with the size of the bytecode generated from the compiled contract source code. Table 3 presents the measurements of core smart contract deployment gas costs, including dataset management and data trading management contracts. Deployment cost is expressed in Gwei, Ether, and USD, with a gas price of 1 ETH = $1.283.23 (2022.12.09). Although gas prices fluctuate, we used this price for simplicity in our evaluation. Our smart contracts were optimized for cost-effectiveness, with the deployment cost for *DatasetMgr.sol* and *DataTradingMr.sol* smart contracts being approximately 0.00234 Ether ($3.03) and 0.0052 Ether ($6.73), respectively.

Table 3. Experimental environment setup and performance metrics.

No	Smart Contract	Deployment Cost		
		Gas Used (Gwei)	ETH	USD †
(1)	DatasetMgr.sol [1]	2,361,016	0.00236102	3.029726562
(2)	DataTradingMgr.sol [2]	5,243,074	0.00524307	6.728069849

[1] Deployed contract address: 0x0d77e6a61c7fb5af4c0c332a524e8b70619f0e49 [2] Deployed contract address: 0x1d682c7098cd34748990925b1bc1651c9cff91eb † ETH Price: 1 ETH = $ 1283.23 (9 December 2022)—https://coinmarketcap.com/.

- *Execution cost*: The execution cost is the overall computational gas cost to execute transaction operations using an EVM, as defined in Equation (12).

$$\varepsilon_{cost} = \sum_{i=1}^{n} OP_i \tag{12}$$

- *Transaction cost*: Transaction costs, also referred to as gas fees or the amount of gas used (G_{cost}), are fees paid by users to miners for processing transactions and ensuring the security of the the blockchain network [27,52]. These fees play a crucial role in resource allocation and network stability. The transaction gas fee (T_{fee}) is calculated by multiplying the amount of gas used G_u by the gas price G_p, as shown in Equation (13). It is worth noting that transaction costs on the blockchain can fluctuate because of factors such as network congestion, gas prices, and the complexity of the smart contract functions involved in the transaction [52]. Table 4 provides a summary of the operational transaction execution gas costs in Gwei, Ether, and USD. The experimental results indicate that, on average, the transaction execution cost is 0.0001 Ether (0.14 USD) for writing operations and 0.000029 Ether (0.04 USD) for reading operations. Tasks that primarily involve retrieving data (getter functions) generally incur lower fees because they do not change the state of the ledger. Conversely, tasks that involve frequent data updates (setter functions) are likely to incur higher costs depending on their complexity.

Table 4. Operational gas costs of core smart contract-related functions.

No	Smart Contract	Operational Cost		
		Gas Used (Gwei)	ETH	USD †
(1)	newDataSet	406,619	0.00040662	0.52178570
(2)	getDatasetDetail	53,166	0.00005317	0.06822421
(3)	isOpenForSale	25,499	0.00002550	0.03272108
(4)	openForSale	48,897	0.00004890	0.06274610
(5)	isSold	25,478	0.00002548	0.06274610
(6)	closeSale	26,953	0.00002695	0.03458690
(7)	getDataPrice	29,117	0.00002912	0.03736381
(8)	changePrice	44,198	0.00004420	0.05671620
(9)	getOwner	25,493	0.00002549	0.03271338
(10)	changeOwnership	44,155	0.00004416	0.05666102
(11)	newOrder	287,227	0.00028723	0.36857830
(12)	getOrderDetail	41,097	0.00004110	0.05273690

Table 4. *Cont.*

No	Smart Contract	Operational Cost		
		Gas Used (Gwei)	ETH	USD †
(13)	getOrderAmount	29,096	0.00002910	0.03733686
(14)	isPaid	25,454	0.00002545	0.03266334
(15)	payOrder	48,800	0.00004880	0.06262162
(16)	confirmReception	36,633	0.00003663	0.04700856
(17)	sendPayment	74,048	0.00007405	0.09502062
(18)	isRefunded	25,500	0.00002550	0.03272237
(19)	refundOrder	76,039	0.00002550	0.03272237

Table 4. *Cont.*

No	Smart Contract	Operational Cost		
		Gas Used (Gwei)	ETH	USD †
(22)	withdraw	35,185	0.00007405	0.09502062
(21)	getBalanceOf	22,234	0.00002223	0.02853134

† ETH Price: 1 ETH = $ 1283.23 (9 December 2022)—https://coinmarketcap.com/.

$$T_{fee} = G_u * G_p \tag{13}$$

4.3. Blockchain and IPFS Performance

Using the PoA consensus protocol, new blocks are added to the Ethereum blockchain every 12s. A block is identified by an index that corresponds to the block number and height of the blockchain. It also contains several parameters, including: the *timestamp*, which indicates when the block was proposed; the *transaction number*, which reflects the number of transactions included in the block; the *size, which* denotes the data size in bytes; the *gas limit*, which represents the maximum gas set by the transactions in the block; and the *gas used*, which is the total units of gas used by transactions in the block. Figure 6a provides measurements of block size per index for 120 sampled blocks, with an average block size of approximately 102.34 KB. The number of transactions per block varies depending on several factors, including the block size limit, the rate at which blocks are produced, and the number of transactions the network processes per unit of time. In Figure 6b, we assessed the number of transactions included in every block, which is 60 on average.

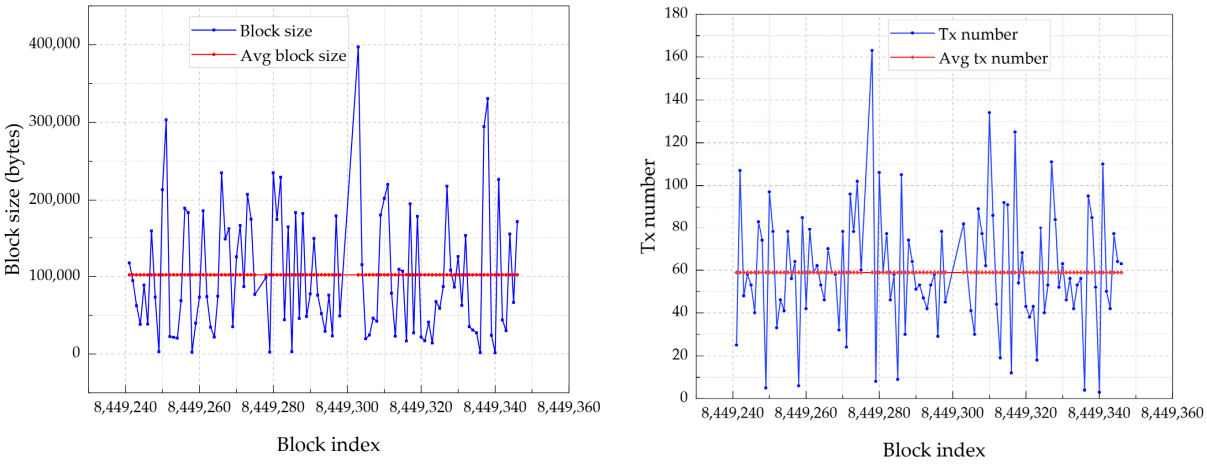


Figure 6. Block size and number of transactions per block index: (**a**) Block size measurements per index for 120 blocks; (**b**) Number of transaction measurements per index for 120 blocks.

Figure 7a shows the gas usage for transactions and blocks processed by the network, with the average block creation and confirmation depending on the gas price. Notably, contracts that consume gas contribute significantly to network usage, with an average of 15,736,700 gas units used per block, which corresponds to the total of gas used by transactions included in the block. Figure 7b shows the relationship between block timestamps, epochs, and slot time variations. The *epoch* parameter indicates the epoch in which the block was proposed, and the *slot* corresponds to the slot in which the block was proposed. Slots refer to opportunities for block creation (adding one valid block). The block timestamp and slot scaled linearly as the block index increased, whereas the block epoch scaled gradually. This resulted because several slots existed in this epoch. The average block time spans approximately 15 s.

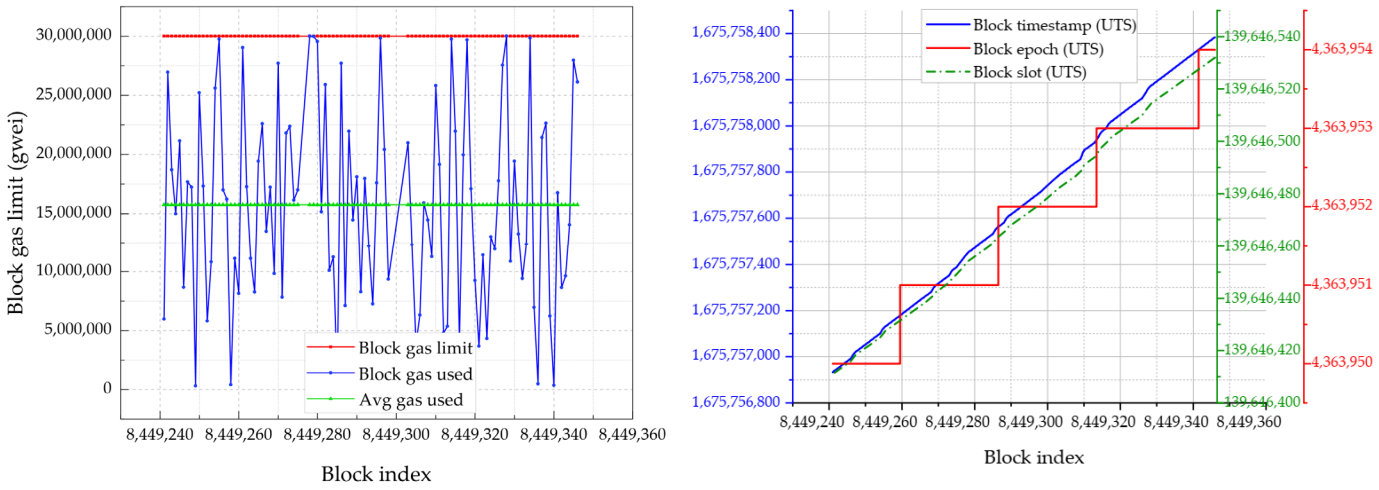


Figure 7. Gas spent, block timestamp, slot, and epoch time per block index: (**a**) Gas limit and gas spent measurements per index for 120 blocks; (**b**) Block timestamp, slot, and epoch time measurements per index for 120 blocks.

Transaction throughput is the number of transactions processed per unit of time, typically measured in transactions per second (TPS). In contrast, transaction latency represents the time taken for a transaction to be processed and included in a block. These metrics in a blockchain network depend on various factors such as network usage, block size, time, and network congestion. As shown in Table 2, the blockchain network exhibited an average transaction throughput and latency of 24 TPS and 36 s respectively, considering the need for at least three block confirmations. Furthermore, the average encryption and storage times per *DPP* object of 38 KB size were 25.4 and 38.6 ms, respectively. The experimental evaluations were performed on a server with an 11th Generation Intel(R) Core^{TM}i7-11700 processor (2.50GHz), 64 GB of RAM, and Gigabit Ethernet network interfaces.

$$B_u = \frac{ADT}{MTR * TP} \tag{14}$$

The IPFS network bandwidth use (B_u) is the ratio of the amount of data being transferred per unit of time to the maximum data transfer rate, as computed using Equation (14). *ADT* denotes the amount of data being transferred that corresponds to the sum of incoming (B_{in}, in bandwidth) and outgoing (B_{out}, out bandwidth) data transfer rates, as expressed in Equation (15). *MTR* denotes the maximum transfer rate achievable for the network, and *TP* is the duration of data transfer.

$$ADT = B_{in} + B_{out} \tag{15}$$

Figure 8 presents the global IPFS network traffic bandwidth use measurements over time demonstrating the network's strong stability and excellent performance with minimal bandwidth consumption. Specifically, the total data transferred comprises 1.2 GB (incoming) and 679 MB (outgoing), with average data rates of 109.70 KB/s (in) and 83.12 KB/s (out).

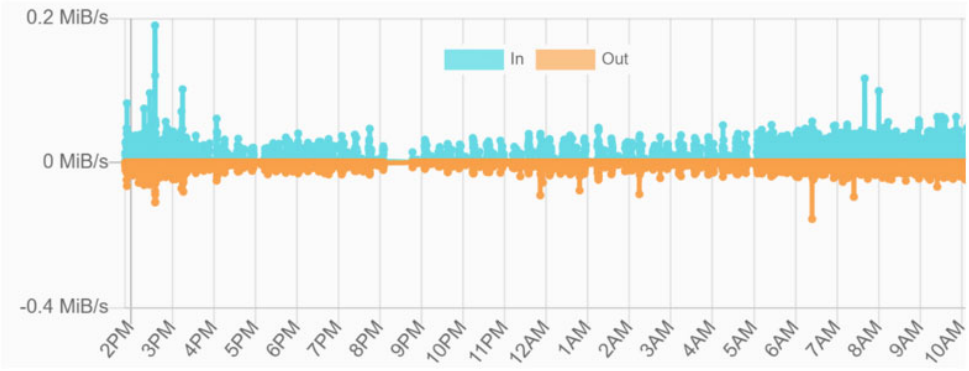


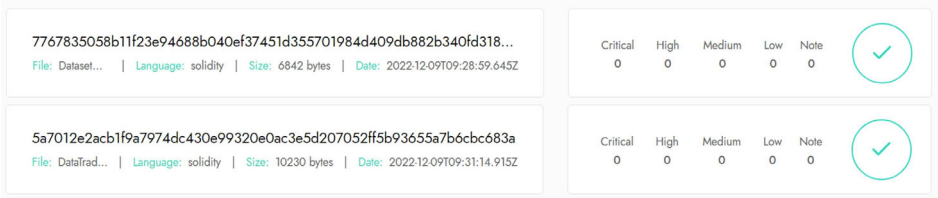
Figure 8. IPFS network traffic in-out-bandwidth utilization measurements.

4.4. Security Analysis

This section assesses the security properties of the proposed model, encompassing the following:

- *Entity and data origin authentication*: Participant entities are identified by Ethereum EOAs, which are authenticated using public-private key pairs and digital signatures [27,52]. Each party possesses a unique private key that is used to generate digital signatures. Verification of the digital signature using the corresponding public key allows the receiving party to authenticate the sender's identity. The data origin is authenticated and verified through ownership and signature proofs stored on the blockchain.
- *Data confidentiality and security*: Blockchain and IPFS technologies ensure data integrity and non-repudiation security [27,29,52]. Digital signatures uphold the integrity of the traded data; any alteration to the data would invalidate the signature [52,53]. In addition, they ensure non-repudiation, preventing the sender from denying sending the data. Once signed and transmitted, the signature serves as an irrefutable proof of consent. However, as a public blockchain, neither Ethereum nor IPFS provides data confidentiality. To address this issue, the data are encrypted before storage in IPFS, while the metadata are stored in the blockchain. The proposed system mitigates replay attacks in which the same signature is repeatedly used ("replayed") for unauthorized actions [52,62]. For example, a payee resubmitting a signature to claim a second payment (double spending attack) poses a serious security risk [62,63]. The security model of the PoA consensus algorithm integrates digital signatures and a tamper-proof ledger to ensure that historical records remain untampered with [54]. This prevents malicious actors from forging payments or falsely reporting asset transfers.
- *Accountability, transparency, and fairness*: Each participant is held accountable for their actions through activity history recorded in a cryptographically secure and tamper-proof blockchain-based ledger [22,26,27]. This ledger serves as an immutable record that fosters transparency and traceability for every transaction. It ensures that participants' actions are transparent and traceable, thus fostering a culture of accountability [40,47–51]. To prevent data misuse and maintain fairness, trade history—encompassing ownership transfers and usage terms—is securely stored on the blockchain. This approach facilitates continuous monitoring and auditing of activities, promoting fairness in the data exchange process. Furthermore, the blockchain-based decentralized P2P storage network enhances transparency, prevents fraud, fosters trust among participants, and promotes ethical behavior within the data marketplace ecosystem.

- *High availability and reliability*: The integration of blockchain and IPFS networks with our proposed system guarantees censorship resistance and robustness, ensuring globally consistent availability and high reliability [29,54–58]. However, the gas fee required to perform transactions introduces a risk of insufficient gas [62–65], which may affect users' ability to operate within the system. To mitigate this issue, users are advised to maintain a sufficient balance in their system operating accounts.
- *Smart contract vulnerabilities*: Ensuring that smart contracts do not contain bugs or security flaws is crucial before their deployment on the blockchain because they cannot be patched or modified once they are deployed [52,63,66]. Our developed *DatasetMgr.sol*[3] and *DataTradingMgr.sol*[4] smart contracts were successfully analyzed using SOOHOOdin[5], a state-of-the-art smart contract security vulnerability analyzer. As shown in Figure 9, they are robust against up-to-date smart contract vulnerabilities, including reentrancy, integer overflow/underflow, unchecked external calls, unprotected ether withdrawal, and gas limit issues [62–66].



[3] https://odin.sooho.io/result/46bcb2f0be2f6eb08f1d9f2eca275392
[4] https://odin.sooho.io/result/47a262b83a652e878b8ac48d9682af6f
[5] https://odin.sooho.io/ (accessed on 30 October 2023)

Figure 9. Smart contract security vulnerability analysis reports.

Next, we present a theoretical analysis of the attack scenarios and mitigation strategies using the proposed framework.

- *Reentrancy attacks*: Malicious actors may exploit reentrancy vulnerabilities in smart contracts to repeatedly invoke functions, potentially withdrawing funds or causing unexpected behavior [62,63]. With the proposed model, such reentrancy attacks can be mitigated by implementing secure coding practices, e.g., using the "*Checks-effects-interactions*" pattern [63–66] to ensure that state changes are made before interacting with external contracts or transferring funds.
- *Front-running attacks*: Malicious actors can execute front-running attacks by monitoring pending transactions and strategically submitting their transactions to exploit price changes or manipulate the of transaction order [52,67]. To mitigate such attacks with the proposed model, mechanisms such as commit-reveal schemes or encrypted order submissions can be implemented to obscure transaction details until execution on the blockchain, preventing attackers from preempting legitimate transactions.
- *Smart contract bugs*: Bugs or logical vulnerabilities in smart contracts can be exploited by attackers to bypass access controls, manipulate data, or cause unexpected behavior in the system [64–66]. These risks can be mitigated through comprehensive code reviews, the use of formal verification techniques, and the introduction of bug bounty programs that incentivize security researchers to identify and report vulnerabilities.
- *Sybil Attacks*: In a Sybil attack, malicious entities create multiple fake identities to gain control over a significant portion of network resources and influence system behavior [68]. To mitigate Sybil attacks, robust identity verification mechanisms, such as proof-of-individuality protocols or reputation systems, can be implemented to prevent the propagation of fake identities and maintain network integrity.

5. Discussion

In this section, we discuss the potential limitations of the proposed approach and the remaining open challenges, including the following:

- *Transaction gas fees*: Transaction fees, particularly in blockchain networks such as Ethereum, pose a major challenge because of their unpredictability and volatility [27,52]. During periods of heightened network activity, gas fees surge as miners compete for limited block space. Consequently, transaction costs escalate, affecting user experience and increasing the risk of transactions running out of gas mid-execution. Such occurrences can result in transaction failures or partial execution, leading to resource wastage and frustrating user experiences [62–65]. Effective mitigation strategies involve meticulous estimation of gas limits and costs before transaction execution and optimization of smart contract code to ensure efficient gas usage [63,66].
- *Smart contract legal design and upgradability considerations*: Smart contract design should include dispute resolution considerations covered by legal regulations. In contrast to traditional software, smart contracts lack upgradability once they are deployed on the blockchain [52,63–65]. This limitation poses challenges in adapting to evolving legal frameworks, particularly in e-commerce and related marketplaces. Although smart contracts enhance efficiency and transparency, they also present the challenge of adapting to unforeseen circumstances [32,64]. Predefined terms can restrict their flexibility, leading to suboptimal outcomes in dynamic trading environments. Developing smart contract-driven reliable and fair protocols requires meticulous coding and a deep understanding of the underlying business logic to minimize loopholes or unintended consequences [63,64]. External factors such as market manipulation or regulatory changes exacerbate smart contract limitations. Addressing these concerns requires innovative smart contract design to improve adaptability and resilience to external influences. Thus, exploring complementary mechanisms such as DApps, rigorous testing, and auditing processes to mitigate potential risks remains an area of future research interest.
- *P2P data marketplace governance*: The absence of a central authority in the proposed P2P data marketplace system necessitates efficient and reliable decentralized governance and trading moderation protocols. Challenges arise from the lack of global data protection regulations, varying jurisdictions, and governance structures governing data sovereignty in P2P data marketplaces.
- *Data quality assessment methods*: Optimizing data quality is essential for pricing and trade reliability [69,70]. Implementing reliable and efficient data quality assessment techniques is imperative for optimizing the quality of the traded data in the proposed system model.
- *Pricing model*: Although this study employs a negotiated price approach between data sellers and buyers, dynamic pricing models such as auction-based [22,49], data quality-based [69,70], or market-driven [71,72] approaches are viable alternatives.
- *Decentralized review, trust, and reputation management schemes*: Efficient decentralized review, trust, and reputation management schemes are essential to leverage the proposed system operating in a trustless environment [73–78].
- *Efficient incentive distribution mechanism*: Given that the proposed P2P data marketplace system model relies on community members, implementing novel, secure, and efficient incentive distribution mechanisms is vital [77,78].
- *Computing resource and energy consumption*: The PoA consensus algorithm is renowned for its efficient computing resource and energy consumption scheme [54], which drives its widespread adoption. In contrast, the PoW consensus algorithm is slower and requires considerable computing resources and energy to solve complex mathematical problems [26,27]. In addition, IPFS network nodes exhibit low computation overhead and bandwidth consumption.
- *Network scalability*: The system's scalability depends on the underlying blockchain and IPFS networks, both grappling with scalability challenges [27–29,55,56]. Balancing scalability, decentralization, and security is essential. Addressing these challenges involves improvements in blockchain scalability and smart contract code optimization, as well as the development of layer 2 scaling solutions and protocol upgrades.

These measures mitigate network congestion, reduce gas fees, and enhance network efficiency and throughput.Top of Form.

6. Conclusions and Future Work

This paper proposes DataMesh+, an innovative, secure, and reliable P2P data exchange model for decentralized self-sovereign data marketplaces, which is underpinned by a robust infrastructure powered by blockchain and decentralized storage technologies. In this model, smart contracts autonomously execute transactions based on pre-agreed terms between buyers and sellers. DataMesh+ represents not only an advancement of the data mesh architectural framework but also a paradigm shift in the operational dynamics of data exchange within marketplaces. This approach differs from traditional data marketplaces in that it prioritizes user-oriented, fair, transparent, secure, and reliable transactions. It enables trustworthy and auditable exchanges between anonymous parties worldwide without relying on centralized trusted third parties. Users retain control over their traded data assets using strong cryptographic techniques, including public-private key pairs and digital signatures, to ensure authentication and accountability. All transactions are recorded in a blockchain ledger, which is known for its tamper-proof and immutable properties, thereby guaranteeing transparency and traceability. Data confidentiality is also prioritized and ensured by robust encryption methods. Furthermore, this paper provides a comprehensive review of the literature on decentralized, blockchain-based P2P data marketplaces. This study explores background research, design considerations, operating principles, and challenges in implementing such systems. The feasibility of the proposed model is demonstrated through a prototype supported by experimental testing and validation, which confirms its reliability and effectiveness. Looking forward, several areas warrant future research. These include the development of reliable methods for assessing data quality, dynamic pricing models, decentralized systems for managing trust and reputation, and efficient governance and incentive distribution protocols for P2P data marketplaces. In addition, empirical studies or surveys to further assess user perception of how the proposed approach enhances user experience and data sovereignty should be conducted. These areas represent crucial steps in the evolution of decentralized data trading platforms, potentially enhancing their efficacy and application.

7. Patents

1. In, H.P.; Merlec, M. M. Blockchain-based safe and reliable data transaction method, and data transaction platform providing system. *WIPO (PCT)*, WO2022145679A1, 7 July 2022.
2. In, H.P.; Merlec, M. M. Blockchain-based secure and trusted data trading methods and platform system. *Korean Patent*, KR102540415B1, 5 June 2023.

Author Contributions: Conceptualization, M.M.M. and H.P.I.; methodology, M.M.M. and H.P.I.; software, M.M.M.; validation, M.M.M. and H.P.I.; formal analysis, M.M.M. and H.P.I.; investigation, M.M.M. and H.P.I.; resources, M.M.M. and H.P.I.; data curation, M.M.M.; writing—original draft preparation, M.M.M.; writing—review and editing, M.M.M. and H.P.I.; visualization, M.M.M.; supervision, H.P.I.; project administration, M.M.M.; funding acquisition, H.P.I. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by the Korea University funding; the National Research Foundation of Korea (NRF), a grant funded by the MSIT (Ministry of Science and ICT) of the Korean government, No. NRF–2021R1A2C2012476 (Blockchain Technology Research for Personal Data Right Assurance); and the Technology Incubator Program for Startup (TIPS) Program (S3306708), funded by the Ministry of Small and Medium Enterprises and Startups (MSS, Korea).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: Author Merlec M.M. was employed by the Department of Computer Science and Engineering, Korea University. Author In H.P. was employed by the Department of Computer Science and Engineering, Korea University and the DAO Solution, Seoul, South Korea. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. IDC; Statista Inc. Volume of Data/Information Created, Captured, Copied, and Consumed Worldwide from 2010 to 2020, with Forecasts from 2021 to 2025 (in Zettabytes). 2021. Available online: https://www.statista.com/statistics/871513/worldwide-data-created/ (accessed on 10 December 2023).
2. Statista Inc. Fortune Business Insights. *Size of The Big Data Analytics Market Worldwide from 2021 to 2029 (in Billion U.S. Dollars)*. 2022. Available online: https://www.statista.com/statistics/1336002/big-data-analytics-market-size/ (accessed on 10 December 2023).
3. Kai, K.; Poikola, A.; Honko, H. Mydata a Nordic Model for Human-Centered Personal Data Management and Processing. 2015. Available online: https://julkaisut.valtioneuvosto.fi/bitstream/handle/10024/78439/MyData-nordic-model.pdf (accessed on 10 December 2023).
4. Cooper, B.F.; Garcia-Molina, H. Peer-to-peer data trading to preserve information. *ACM Trans. Inf. Syst.* **2002**, *20*, 133–170. [CrossRef]
5. Spiekermann, M. Data marketplaces: Trends and monetisation of data goods. *Intereconomics* **2019**, *54*, 208–216. [CrossRef]
6. Cao, T.D.; Pham, T.V.; Vu, Q.H.; Truong, H.L.; Le, D.H.; Dustdar, S. MARSA: A marketplace for realtime human sensing data. *ACM Trans. Internet Technol.* **2016**, *16*, 1–21. [CrossRef]
7. Parra-Arnau, J. Optimized, direct sale of privacy in personal data marketplaces. *Info. Sci.* **2018**, *424*, 354–384. [CrossRef]
8. Oh, H.; Park, S.; Lee, G.M.; Heo, H.; Choi, J.K. Personal Data Trading Scheme for Data Brokers in IoT Data Marketplaces. *IEEE Access* **2019**, *7*, 40120–40132. [CrossRef]
9. Hatamian, M. Technological Barriers of (non)Blockchain Enabled IoT Data Marketplaces. In Proceedings of the 2021 25th International Computer Science and Engineering Conference (ICSEC), Chiang Rai, Thailand, 16 November 2021; pp. 39–44.
10. Fruhwirth, M.; Rachinger, M.; Prlja, E. Discovering Business Models of Data Marketplaces. In Proceedings of the 53rd Hawaii International Conference on System Sciences, Maui, HI, USA, 7–10 January 2020; pp. 5738–5747.
11. Van de Ven, M.; Abbas, A.E.; Kwee, Z.; de Reuver, M. Creating a Taxonomy of Business Models for Data Marketplaces. In Proceedings of the 34th Bled eConference-Digital Support from Crisis to Progressive Change, Online, 27–30 June 2021; pp. 313–325.
12. de la Vega, F.; Soriano, J.; Jimenez, M.; Lizcano, D. A peer-to- peer architecture for distributed data monetization in fog computing scenarios. *Wirel. Commun. Mob. Comput.* **2018**, *2018*, 5758741. [CrossRef]
13. Sabounchi, M.; Wei, J. Blockchain-enabled peer-to-peer data trading mechanism. In Proceedings of the 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Halifax, NS, Canada, 30 July–3 August 2018; pp. 1410–1416.
14. Serrano, N.; Cuenca, F. A Peer-to-Peer Ownership-Preserving Data Marketplace. In Proceedings of the 2021 IEEE International Conference on Blockchain (Blockchain), Melbourne, Australia, 6–8 December 2021; pp. 394–400.
15. Klems, M.; Eberhardt, J.; Tai, S.; Härtlein, S.; Buchholz, S.; Tidjani, A. Trustless intermediation in blockchain-based decentralized service marketplaces. In Proceedings of the Service-Oriented Computing: 15th International Conference, ICSOC 2017, Malaga, Spain, 13–16 November 2017; pp. 731–739.
16. Abbas, A.E.; Agahari, W.; van de Ven, M.; Zuiderwijk, A.; de Reuver, M. Business Data Sharing through Data Marketplaces: A Systematic Literature Review. *J. Theor. Appl. Electron. Commer. Res.* **2021**, *16*, 3321–3339. [CrossRef]
17. Andres, S.A.; Laoutaris, N. A survey of data marketplaces and their business models. *SIGMOD Rec.* **2022**, *51*, 18–29.
18. Abbas, A.E. Designing Data Governance Mechanisms for Data Marketplace Meta-Platforms. In Proceedings of the 34th Bled eConference–Digital Support from Crisis to Progressive Change, Online, 27–30 June 2021; pp. 695–707.
19. De Capitani di Vimercati, S.; Foresti, S.; Livraga, G.; Samarati, P. Toward owners' control in digital data markets. *IEEE Syst. J.* **2021**, *15*, 1299–1306. [CrossRef]
20. Merlec, M.M.; Lee, Y.K.; Hong, S.-P.; In, H.P. A Smart Contract-Based Dynamic Consent Management System for Personal Data Usage under GDPR. *Sensors* **2021**, *21*, 7994. [CrossRef]
21. Nasonov, D.; Visheratin, A.A.; Boukhanovsky, A. *Blockchain-Based Transaction Integrity in Distributed Big Data Marketplace*; Springer International Publishing: Cham, Switzerland, 2018; pp. 569–577.
22. Wang, H.; Qin, H.; Zhao, M.; Wei, X.; Shen, H.; Susilo, W. Blockchain-based fair payment smart contract for public cloud storage auditing. *Inf. Sci.* **2020**, *519*, 348–362. [CrossRef]
23. Voigt, P.; von dem Bussche, A. *The EU General Data Protection Regulation (GDPR), A Practical Guide*; Springer International Publishing AG: Cham, Switzerland, 2017.
24. Dehghani, Z. How to Move Beyond a Monolithic Data Lake to a Distributed Data Mesh. 2019. Available online: https://martinfowler.com/articles/data-monolith-to-mesh.html (accessed on 10 December 2023).
25. Dehghani, Z. *Data Mesh: Delivering Data-Driven Value at Scale*; O'Reilly Media: Sebastopol, CA, USA, 2022.

26. Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Bus. Rev.* **2008**, 21260.
27. Wood, G. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Proj. Yellow Pap.* **2014**, *151*, 1–32.
28. Xu, X.; Weber, I.; Staples, M.; Zhu, L.; Bosch, J.; Bass, L.; Pautasso, C.; Rimba, P. A taxonomy of blockchain-based systems for architecture design. In Proceedings of the 2017 IEEE International Conference on Software Architecture (ICSA), IEEE, Gothenburg, Sweden, 3 April 2017; pp. 243–252.
29. Benet, J. Ipfs-content addressed, versioned, p2p file system. *arXiv* **2014**, arXiv:1407.3561.
30. Machado, I.A.; Costa, C.; Santos, M.Y. Data Mesh: Concepts and Principles of a Paradigm Shift in Data Architectures. *Procedia Comput. Sci.* **2022**, *196*, 263–271. [CrossRef]
31. Majchrzak, J.; Balnojan, S.; Siwiak, M. *Data Mesh in Action*; Simon and Schuster: New York, NY, USA, 2023.
32. Khan, S.N.; Loukil, F.; Ghedira-Guegan, C.; Benkhelifa, E.; Bani-Hani, A. Blockchain smart contracts: Applications, challenges, and future trends. *Peer Peer Netw. App.* **2021**, *14*, 2901–2925. [CrossRef]
33. Subramanian, H. Decentralized blockchain-based electronic marketplaces. *Comm. of the ACM* **2017**, *61*, 78–84. [CrossRef]
34. Kabi, O.R.; Franqueira, V.N.L. Blockchain-Based Distributed Marketplace. In *Business Information Systems Workshops*; Abramowicz, W., Paschke, A., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 197–210.
35. Weber, T.; Prinz, W. Trading User Data: A Blockchain Based Approach. In Proceedings of the 2019 Sixth International Conference on IoT: Systems, Management and Security (IOTSMS), Granada, Spain, 22–25 October 2019; pp. 547–554.
36. Lawrenz, S.; Sharma, P.; Rausch, A. Blockchain technology as an approach for data marketplaces. In Proceedings of the 2019 International Conference on Blockchain Technology, Honolulu, HI, USA, 15 March 2019; pp. 55–59.
37. Hyunkyung, Y.; Ko, N. Blockchain based data marketplace system. In Proceedings of the 2020 International Conference on Information and Communication Technology Convergence (ICTC), IEEE, Jeju, Republic of Korea, 21–23 October 2020; pp. 1255–1257.
38. Banerjee, P.; Ruj, S. Blockchain Enabled Data Marketplace–Design and Challenges. *arXiv* **2018**, arXiv:1811.11462.
39. Li, J.; Grintsvayg, A.; Kauffman, J.; Fleming, C. LBRY: A Blockchain-Based Decentralized Digital Content Marketplace. In Proceedings of the 2020 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS), Oxford, UK, 3–6 August 2020; pp. 42–51.
40. Dai, W.; Dai, C.; Choo, K.K.R.; Cui, C.; Zou, D.; Jin, H. SDTE: A secure blockchain-based data trading ecosystem. *IEEE Trans. Inf. Forensics Secur.* **2019**, *15*, 725–737. [CrossRef]
41. Xiong, W.; Xiong, L. Smart contract based data trading mode using blockchain and machine learning. *IEEE Access* **2019**, *7*, 102331–102344. [CrossRef]
42. Hu, D.; Li, Y.; Pan, L.; Li, M.; Zheng, S. A blockchain-based trading system for big data. *Comput. Netw.* **2021**, *191*, 107994. [CrossRef]
43. Miehle, D.; Meyer, M.M.; Luckow, A.; Bruegge, B.; Essig, M. Toward a decentralized marketplace for self-maintaining machines. In Proceedings of the 2019 IEEE International Conference on Blockchain (Blockchain), Atlanta, GA, USA, 14 July 2019; pp. 431–438.
44. Christidis, J.; Karkazis, P.A.; Papadopoulos, P.; Leligou, H.C. Decentralized Blockchain-Based IoT Data Marketplaces. *J. Sens. Actuator Netw.* **2022**, *11*, 39. [CrossRef]
45. Suliman, A.; Husain, Z.; Abououf, M.; Alblooshi, M.; Salah, K. Monetization of IoT data using smart contracts. *IET Netw.* **2019**, *8*, 32–37. [CrossRef]
46. Gupta, P.; Kanhere, S.; Jurdak, R. A decentralized IoT data marketplace. *arXiv* **2019**, arXiv:1906.01799.
47. Özyilmaz, K.R.; Doğan, M.; Yurdakul, A. IDMoB: IoT data marketplace on blockchain. In Proceedings of the 2018 Crypto Valley Conference on Blockchain Technology (CVCBT), Zug, Switzerland, 20–22 June 2018; pp. 11–19.
48. Bajoudah, S.; Dong, C.; Missier, P. Toward a decentralized, trust-less marketplace for brokered IoT data trading using blockchain. In Proceedings of the 2019 IEEE International Conference on Blockchain (Blockchain), Atlanta, GA, USA, 14–17 July 2019; pp. 339–346.
49. Chen, C.; Wu, J.; Lin, H.; Chen, W.; Zheng, Z. A secure and efficient blockchain-based data trading approach for internet of vehicles. *IEEE Trans. Veh. Technol.* **2019**, *68*, 9110–9121. [CrossRef]
50. Zichichi, M.; Ferretti, S.; Rodríguez-Doncel, V. Decentralized Personal Data Marketplaces: How Participation in a DAO Can Support the Production of Citizen-Generated Data. *Sensors* **2022**, *22*, 6260. [CrossRef] [PubMed]
51. Travizano, M.; Sarraute, C.; Dolata, M.; French, A.M.; Treiblmaier, H. Wibson: A case study of a decentralized, privacy-preserving data marketplace. In *Blockchain and Distributed Ledger Technology Use Cases*; Springer: Cham, Switzerland, 2020; pp. 149–170.
52. Antonopoulos, A.M.; Wood, G. *Mastering Ethereum: Building Smart Contracts and Dapps*; O'Reilly Media: Sebastopol, CA, USA, 2018.
53. Johnson, D.; Menezes, A.; Vanstone, S. The Elliptic Curve Digital Signature Algorithm (ECDSA). *IJIS* **2001**, *1*, 36–63. [CrossRef]
54. Islam, M.M.; Merlec, M.M.; In, H.P. A comparative analysis of proof-of-authority consensus algorithms: Aura vs Clique. In Proceedings of the 2022 IEEE International Conference on Services Computing (SCC), Barcelona, Spain, 10–16 July 2022; pp. 327–332.
55. Naz, M.; Al-zahrani, F.A.; Khalid, R.; Javaid, N.; Qamar, A.M.; Afzal, M.K.; Shafiq, M. A secure data sharing platform using blockchain and interplanetary file system. *Sustainability* **2019**, *11*, 7054. [CrossRef]
56. Merlec, M.M.; Islam, M.M.; Lee, Y.K.; In, H.P. A consortium blockchain-based secure and trusted electronic portfolio management scheme. *Sensors* **2022**, *22*, 1271. [CrossRef]

57. Cruz, J.P.; Kaji, Y.; Yanai, N. RBAC-SC: Role-Based Access Control Using Smart Contract. *IEEE Access* **2018**, *6*, 12240–12251. [CrossRef]
58. Xu, R.; Chen, Y.; Blasch, E. Decentralized Access Control for IoT Based on Blockchain and Smart Contract. In *Modeling and Design of Secure Internet of Things*; Wiley: Hoboken, NJ, USA, 2018; pp. 505–528.
59. Bertoni, G.; Daemen, J.; Peeters, M.; Van Assche, G. The KECCAK Reference, Version 3.0, January 2011. Available online: http://keccak.noekeon.org/Keccak-reference-3.0.pdf (accessed on 10 December 2023).
60. Blockscan Chat. Available online: https://chat.blockscan.com/ (accessed on 10 December 2023).
61. Zhang, X.; Hsiao, H.; Hasker, G.; Chan, H.; Perrig, A.; Andersen, D. Scion: Scalability, control, and isolation on next-generation networks. In Proceedings of the 2011 IEEE Symposium on Security and Privacy, Oakland, CA, USA, 22–25 May 2011; pp. 212–227.
62. Atzei, N.; Bartoletti, M.; Cimoli, T. A survey of attacks on ethereum smart contracts (sok). In Proceedings of the International Conference on Principles of Security and Trust, Uppsala, Sweden, 22–29 April 2017; pp. 164–186.
63. Diligence, ConsenSys. Ethereum Smart Contract Best Practices-Known Attacks. 2018. Available online: https://ethereum-contract-security-techniques-and-tips.readthedocs.io/en/latest/ (accessed on 10 December 2023).
64. Huang, Y.; Bian, Y.; Li, R.; Zhao, J.L.; Shi, P. Smart contract security: A software lifecycle perspective. *IEEE Access* **2019**, *7*, 150184–150202. [CrossRef]
65. Kushwaha, S.S.; Joshi, S.; Singh, D.; Kaur, M.; Lee, H.N. Systematic review of security vulnerabilities in Ethereum blockchain smart contract. *IEEE Access* **2022**, *10*, 6605–6621. [CrossRef]
66. Sayeed, S.; Marco-Gisbert, H.; Caira, T. Smart contract: Attacks and protections. *IEEE Access* **2020**, *8*, 24416–24427. [CrossRef]
67. Eskandari, S.; Moosavi, S.; Clark, J. *SoK: Transparent Dishonesty: Front-Running Attacks on Blockchain*; Springer International Publishing: New York, NY, USA, 2020; Volume 11599, pp. 170–189.
68. Zhang, K.; Liang, X.; Lu, R.; Shen, X. Sybil Attacks and Their Defenses in the Internet of Things. *IEEE Internet Things J.* **2014**, *1*, 372–383. [CrossRef]
69. Liang, F.; Yu, W.; An, D.; Yang, Q.; Fu, X.; Zhao, W. A survey on big data market: Pricing trading and protection. *IEEE Access* **2018**, *6*, 15132–15154. [CrossRef]
70. Yang, J.; Zhao, C.; Xing, C. Big data market optimization pricing model based on data quality. *Complexity* **2019**, *2019*, 5964068. [CrossRef]
71. Azcoitia, S.A.; Iordanou, C.; Laoutaris, N. Measuring the price of data in commercial data marketplaces. In Proceedings of the 1st International Workshop on Data Economy, Rome, Italy, 9 December 2022; pp. 1–7.
72. Li, B.; Wu, M.; Li, Z.; Sun, Y. A pricing model for subscriptions in data transactions. *Connect. Sci.* **2022**, *34*, 529–550. [CrossRef]
73. Avyukt, A.; Ramachandran, G.; Krishnamachari, B. A Decentralized Review System for Data Marketplaces. In Proceedings of the 2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Sydney, Australia, 3–6 May 2021; pp. 1–9.
74. Park, J.S.; Youn, T.Y.; Kim, H.B.; Rhee, K.H.; Shin, S.U. Smart contract-based review system for an IoT data marketplace. *Sensors* **2018**, *18*, 3577. [CrossRef] [PubMed]
75. Putra, G.D.; Dedeoglu, V.; Kanhere, S.S.; Jurdak, R. Blockchain for trust and reputation management in cyber-physical systems. In *Handbook on Blockchain*; Springer: Cham, Switzerland, 2022; pp. 339–362.
76. Khaqqi, K.N.; Sikorski, J.J.; Hadinoto, K.; Kraft, M. Incorporating seller/buyer reputation-based system in blockchain-enabled emission trading application. *Appl. Energy* **2018**, *209*, 8–19. [CrossRef]
77. Abubaker, Z.; Khan, A.U.; Almogren, A.; Abbas, S.; Javaid, A.; Radwan, A.; Javaid, N. Trustful data trading through monetizing IoT data using BlockChain based review system. *Concurr. Comput. Pract. Exper.* **2022**, *34*, e6739. [CrossRef]
78. Rizwan, M.; Sohail, M.N.; Asheralieva, A.; Anjum, A.; Angin, P. SAID: ECC-Based Secure Authentication and Incentive Distribution Mechanism for Blockchain-Enabled Data Sharing System. In Proceedings of the 2021 IEEE International Conference on Blockchain (Blockchain), Melbourne, Australia, 6–8 December 2021; pp. 530–537.