Fbereum: A Novel Distributed Ledger Technology System

Dylan Yu[1], Ethan Yang[2], Alissa Shen[3], Dan Tamir[4](✉), and Naphtali Rishe[5]

[1] Dulles High School, Sugar Land, Texas, USA
[2] Westwood High School, Austin, Texas, USA
[3] St. Stephen's Episcopal School, Austin, Texas, USA
[4] Texas State University, San Marcos, Texas, USA
dt19@txstate.edu
[5] Florida International University, Miami, FL, USA
ndr@acm.org

Abstract. Over the past several years, due to the progression toward data-driven scientific disciplines, the field of Big Data has gained significant importance. These developments pose certain challenges in the area of efficient, effective, and secure management and transmission of digital information. This paper presents and evaluates a novel Distributed Ledger Technology (DLT) system, Fibereum, in a variety of use-cases, including a DLT-based system for Big Data exchange, as well as the fungible and non-fungible exchange of artwork, goods, commodities, and digital currency. Fibereum's innovations include the application of non-linear data structures and a new concept of Lazy Verification. We demonstrate the benefits of these novel features for DLT system applications' cost performance and their added resilience towards cyber-attacks via the consideration of several use cases.

Keywords: Distributed Ledger Technology · Blockchain · Bitcoin · Ethereum · Hyper Ledger · Consensus Verification · Cybersecurity · Proof of Work · Proof of Elapsed Time

1 Introduction

This paper presents Fibereum, a Distributed Ledger Technology (DLT) system that applies novel methods to address several issues with current Blockchain [1–5] data structures and storage mechanisms, including deficiencies with respect to defending against blockchain attacks. A typical implementation of conventional DLT systems is via Blockchain. That is, conventional DLT systems, such as the permissionless Bitcoin [3] and Ethereum [4], as well as the permission-based Hyper Ledger Technology [5], use a linear data structure to manage blocks of data. Furthermore, conventional DLT systems often require a complex time- and energy-consuming process for the verification of the DLT system integrity. The DLT system presented here, Fibereum, utilizes novel methods for Big Data exchange, as well as for the fungible and non-fungible exchange of artwork, goods, commodities, and digital currency.

The novel features of the proposed DLT system include: **(i)** the enablement of the use of non-linear data structures-based systems; **(ii)** employing a procedure of lazy verification, where the verification of the DLT system integrity is delayed indefinitely and applied only on a need-to-do basis; and **(iii)** the enablement of permissionless as well as permission-based implementations. The use of non-linear data structures for the storage of transactions in blocks and storage of blocks within the DLT system enhances the efficiency of the overall system. For example, in one implementation, the proposed DLT system can use cryptographic trees for intra-block and inter-block management in conjunction with lazy verification. This approach significantly improves the management and security of Big Data and other types of digital data-driven systems. Furthermore, the use of lazy verification along with non-linear data structures, as well as the utilization of a time/energy consumption-efficient consensus mechanism, can provide a significant saving in energy consumption.

Fibereum introduces several innovative modifications to Blockchain technology, providing a more general framework for DLT systems. The modifications extend the utility of DLT systems to several new applications, including business-to-business data governance and data exchange. The Fibereum DLT system has various applications in the fields of Big Data, including Healthcare, Transportation, Smart Cities, the Internet of Things, and process control. The Fibereum DLT system is also suitable for applications such as digital currency, smart contracts, licensing [6], inventory management [7], supply chain management [8], counterfeit detection [9], and the exchange of copyrighted material, e.g., Non-Fungible Tokens [10].

We have developed and implemented an event-based simulation for Fibereum use cases. The simulation and theoretical analysis show that, due to the option to use non-liner data structures and the concept of lazy verification, in most cases Fibereum computational complexity is lower than other DLT systems such as the Bitcoin Blockchain.

The rest of this paper is organized as follows. Section 2 provides background information and definitions. Section 3 includes a literature review. Section 4 presents the main features of the Fibereum DLT system. Section 5 presents several Fibereum use-cases, and Sect. 6 includes a conclusion and directions for further research.

2 Background and Definitions

2.1 Definitions

A blockchain is a peer-to-peer network that stores transactions between multiple parties, organized as a cryptographic linked list – i.e., a chain of nodes. Blockchains attempt to guarantee decentralization, transparency, and immutability [1, 2].

A **Merkle tree** is a tree in which all the leaves contain the cryptographic hash of a block of data, potentially along with the data, and every non-leaf node contains the hash of its child nodes' data [11]. The notion of the basic Merkle tree can be extended to Merkle Heaps (Min and Max heaps) [4], Merkle binary search trees [4], Merkle Hash tables [4], and Merkle cyclic and acyclic graphs [4].

A **permissionless** DLT system is open to the public. Any user can create or access data or smart contracts in the DLT system, and all the transactions made on the DLT system are displayed to all the users, making the permissionless DLT system completely transparent [1–3]. In general, blocks are mined by users, referred to as miners, onto the ledger, in exchange for incentives for the miners [1–3]. Furthermore, users may be engaged in establishing the DLT system's integrity (potentially with verification incentives) [1–3]. For specific use-cases, Fibereum offers a permissionless version of a DLT system, where the system construction is extremely simple and does not require significant incentives for miners. Additionally, Fibereum offers an alternative approach for verifying the integrity and incentivizing the verification process. This approach is referred to as 'Lazy Verification.' In this case, initially, the DLT system is in a "verifiable" state. The verification and its incentives are enacted only on a "need to do" basis.

Lazy verification is a form of consensus term-setting, first introduced in the context of Fibereum, where the consensus verification process (as well as verification incentives) is/are delayed as much as possible and only performed when an immediate urgent need, e.g., taking care of an exception, arises. While continuous and prompt verification, which may require ample incentives, computational resources, and a high amount of energy consumption, is mandatory in certain use-cases and applications (e.g., digital currency), lazy verification allows for an efficient method of verifying transactions in a DLT system as it removes the unnecessary steps of verifying every block before an exception has occurred. The verification algorithm may apply the same concept as the standard blockchains' verification procedures, such as the Proof-of-Work (PoW)-based Byzantine consensus [3, 12], but with more scalability for Big Data when there are large amounts of data entering the DLT system at a high rate.

A **permission-based** DLT system is a private network where only certain users are authorized to access the DLT system. The network users are identifiable and complete anonymity is not possible [13, 14]. Hence, access control and encryption may be implemented as a part of the permission mechanism [13]. **Hyperledger Fabric Technology** (HFT) is the most commonly used framework for permission-based blockchains [5]. Fibereum offers a permission-based version of a DLT system.

A **Consensus Algorithm** replaces a centralized authority to preserve the security and fault tolerance of a DLT system. Two consensus algorithms, Proof-of-Work (PoW) [3, 13] and Proof-of-Elapsed-Time (PoET) [14] are most relevant for the Fibereum use-cases. Other commonly used consensus algorithms include Proof-of-Inclusion [15] and Proof-of-Stake [16]. The Practical Byzantine Fault Tolerance is often used as a part of consensus algorithms [12]. We elaborate on the PoET mechanism, which is less known to many DLT system practitioners and is advantageous in terms of cost performance over other consensus mechanisms, especially with respect to some Fibereum use-cases.

PoET is a consensus algorithm in which all nodes "sleep" for an arbitrary amount of time, with the first node to wake up receiving authorization/rewards for verification and mining. PoET is more energy efficient and less resource costly than PoW. Nevertheless, this algorithm must resolve "collisions" in a way that may be similar to the collision detection, avoidance, and resolution of the Carrier Sense, Multiple Access, with Collision Detection (CSMACD) procedures that govern many of the commonly used communication protocols [17, 18].

Notably, Fibereum's use of lazy verification, along with providing the option to use non-linear data structures as well as time/energy consumption efficient DLT system construction and consensus verification mechanisms, can provide improved resilience against attacks, as well as a significant saving in operational costs.

2.2 Resilience and Security

In this sub-section, we list several of the common attacks applied to existing blockchain DLT systems and related security concerns. In Sect. 4, we will refer to these items in the context of Fibereum.

Some of the common attacks on the Bitcoin Blockchain are Eclipse/Sybil attacks [20] and double-spending attacks [19]. Among other attack types are the Vector76 attack [20], the Blockchain reorganization attack [3, 21], and Denial of Service (DoS) attacks [22]. Additionally, careless management of passwords and security measures might jeopardize the anonymity of the DLT system users. The **Majority Attack / 51% attack,** is one of the most commonly discussed attacks in the context of digital coins [3, 21]. In this type of attack, the attacker controls more than 50% of the network's computation power and thus is able to successfully perform bogus blockchain modifications and reorganizations and obtain [temporary] consensus for the bogus blocks [3, 21].

Many of the attacks on the Bitcoin blockchain listed above are applicable to the Ethereum blockchain. An additional set of attacks on Ethereum exploits its smart contract functionality, specifically the computational complexity of the embedded "Turing Complete" [23] functions, which is quantified in terms of "gas," reflecting the cost of computation [4]. The major Ethereum attacks include Reentrancy [24], Front running [25], Integer Overflow and Underflow attacks [26], Unexpected Revert attacks [26], Gas Limit attacks [27], Block Stuffing attacks [27], and Multi-Signature attacks [28].

Common Attacks on HFT Blockchain deal with the centralized and permission-based aspects of the HFT DLT system, particularly attacking the membership service provider that authorizes and provides permissions for entrance into the blockchain and blockchain transactions [29]. The major HFT attacks are the Insider Threat attacks [30] and the Certificate of Authority attack [31].

3 Literature Review

DLT systems provide a decentralized platform. Hence, its potential usage in the field of big data exchange may have significant benefits. Nevertheless, to the best of our knowledge, Fibereum is the first DLT system that provides an optimal solution for that purpose while offering significant benefits in other use-cases. Since Fibereum is a unique DLT system using lazy verification and non-linear data structures, its composition and computation processes are especially efficient. An extensive literature review performed resulted in very few publications that specifically address the issues that Fibereum addresses. Three papers are listed below.

Cäsar et al. have developed a DLT system named Cerberus that focuses on the particular ordering of State Machine Replication (SMR) across a network of unreliable machines [32]. This DLT's consensus mechanism is based on a leader-based Byzantine fault-tolerant consensus approach [12]. In contrast, we propose consensus mechanisms such as the lazy verification mechanism, which minimizes the need for prompt and incentivized consensus, and enhance fault tolerance at lower computational resources.

Snow has developed Factom, a general-purpose data layer that creates a consensus system to ensure that entries are quickly recorded [33]. Comparatively, Fibereum offers the lazy verification approach, which is more suitable for numerous use-cases (see SubSect. 4.2 and Sect. 5). Additionally, Fibereum offers the use of other non-linear data structures, such as Merkle-heaps, for the Inter-block DLT system's construction and maintenance. Thus, Fibereum enables a more efficient method of verification and DLT operation, especially for Big Data exchange.

Parachain [34] uses chains that are processed in parallel, thereby has the potential to improve throughput. Parachain DLT has not addressed certain issues related to overseeing blockchain creation. Fibereum provides better support of "safe" parallelism via the mechanism of using Merkle Heaps for inter-block construction; at the same time, the mechanism is highly efficient and provides $O(log(n))$ [4] complexity for DLT consensus and construction.

Other relevant papers that are not completely overlapping Fibereum concepts and targeted use cases. Examples include work Gay et al. [35], and by Zhu et al. [36].

4 The Fibereum DLT

The Fibereum DLT introduces the following novel features: (i) Options for storing information/transactions blocks in data structures, including Merkle trees, Merkle heaps, or Merkle hash tables, rather than as a linear list in the form of a blockchain; (ii) Enablement of low complexity algorithms and parallel processing; (iii) Lazy Verification – minimizing the need for incentivized DLT systems' construction and consensus verification; (iv) Enablement of permission-based and permissionless modes of access and operations; (v) Enablement of encryption and compression of the data; (vi) Enabling improved cyber security, protection against attacks, and fault tolerance; (vii) Providing additional layers of encryption and digital signatures (in addition to cryptographic hash functions referred to as the digests [1, 2]); (viii) Enablement of Embedded Turing Complete [23], static and/or dynamic, code, which provides efficient management of smart contracts [4] and End User License Agreements (EULA) [35, 36]; (ix) Enablement of efficient management of static, dynamic, and ad-hoc federated data, including terms and policy management for monetization (please see the section on use-cases); (x) Enablement of systems for data governance and currency exchange; (xi) Providing an option for using more than one DLT system in tandem. The latter is referred to as multi-plan implementations. For example, in the exception maintenance use-cases (4.2 and 5.1), we introduce three DLT system plans: one for data transactions governance, one for data exchange, and one for smart contracts – defining data ownership, usage policies, rights, management, and governance.

4.1 A Merkle-Based Verification System

Implementations of Merkle trees-based non-linear DLT systems offer several key advantages over linear blockchains. These advantages include: (i) Merkle trees-based implementations maintain the integrity by cascading any change to the cryptographic hash.

Pointing from the previous node in the tree back to the Merkle root would invalidate the changed block. (ii) Merkle trees-based implementations are typically efficient for the construction and verification of DLT systems, offering the complexity of $O(log(n))$ (or, in some cases, $O(n \times log(n))$) rather than $O(n)$ (or, in some cases, $O(n^2)$). Hence, Merkle tree-based implementations can reduce the temporal and spatial computational complexity. Moreover, without a Merkle tree, the data would need to be sent across the network for verification. Hence, Merkle trees reduce the data transfer delay. (iii) The Merkle tree structure of the local blockchain can use Proof of Inclusion [15], a method of verifying the validity of data without needing to move data across all parts of the network. This algorithm can work in conjunction with consensus mechanisms, such as PoW and PoET. The joint Merkle heap, proposed in some of the Fibereum use-cases, is beneficial for efficient traversal without revealing all portions of the structure. To further demonstrate the principles of Fibereum's operation and its novelty, we present an important use-case here, and the rest of the use-cases are presented in Sect. 5.

4.2 Exception Maintenance 1

This use-case considers the situation that an airplane manufacturing company X buys an engine from an engine manufacturing company Z, and the engine is installed on an airplane of an airline company Y. In other words, this is a business-to-business-to-business (B2B2B) scenario. To simplify the example, we assume that the engine is in X's possession, and the use-case is a typical B2B use-case between X and Y. Generally, Y owns the data. However, in some scenarios, the ownership of the data might be shared between X and Y. It is assumed that according to a licensing agreement between X and Y, Y collects and owns the engine's sensor data. The proposed Fibereum DLT system is designed to be used for managing data usage, ownership, and storage used by the companies and their affiliates in a way that enables dealing with exceptions in the regular operation of the airplane.

DLT System Operation Procedures

(i) Verifiable sensor data is collected in a joint heap, accessible by both X and Y. The data is "verifiable" in the sense that it includes means for verifying its authenticity, e.g., cryptographic hashes of time stamps and sensor IDs. (ii) Each heap node stores specific components of the sensor data (e.g., temperature, pressure, and position) in the storage area. In some implementations, the heap storage is based on a string pool [38]. (iii)Similarly to blockchains, such as the Bitcoin blockchain, the data is stored in blocks and organized in blocks via an internal Merkle tree (the Intra-Merkle Tree). These blocks are maintained by an external Merkle heap (the Inter-Merkle heap). (iv) Inter-block and Intra-block storage via Merkle heaps or trees simply imply that both parties can traverse each piece of sensor data as well as the entire collection of sensor data efficiently and "quickly." (v) If legitimately requested, timely verification is used to ensure data integrity. The fact that the verification is done only on a need-to-do basis and at the time of the need-to-do verification is the origin of the name "lazy verification." The lazy verification process can reduce the cost of operations.

Lazy Verification

The lazy verification process is activated when the need arises (e.g., dealing with an exception in the engine's operation). At this time, a new DLT system based on a data structure such as a sorted Merkle (tree in this example) or a Blockchain (in use-case 5.1) may be constructed. If there is no malicious activity, valid blocks are fetched from the joint heap of the first DLT, and each valid block is appended to the second DLT system in the order of the recency of activities.

Our current implementation of the system has the following components: permissionless or permission-based application of the Fibereum DLT system used for validated exchanges, min-heap DLT system for exchanges that still have to be validated, and license agreement programmed into smart contracts.

To elaborate: data continuously flows from Company X's engine to the joint Merkle heap-based DLT system (DLT System-1) shared by both X and Y. On exception (e.g., overheating, mis-assembly, or exhaustion), X or Y can choose to request lazy verification regarding the exception. In this case, X and Y can nominate one or more proxies and assign the task of verification to the proxies. At this point, the process might resemble verification on blockchains, such as the Bitcoin blockchain. The proxies act like miners and assemble a second DLT system (DLT System-2), which is a Merkle Tree-based DLT or a blockchain-based DLT. DLT System-2 contains only blocks that have been verified by the proxies. Following the request, Companies X and Y (or their proxies) check the sensor data stored in DLT System-2 to determine whether there is a legitimate error, such as engine exhaustion. If there is a nonfunctional component, the lazy verification tags the exception as valid, and Company Y can take steps to fix the issue with the engine. Otherwise, the lazy verification deems the exception invalid, implying that there has been a human error in maintaining the engine and that the engine is functioning properly.

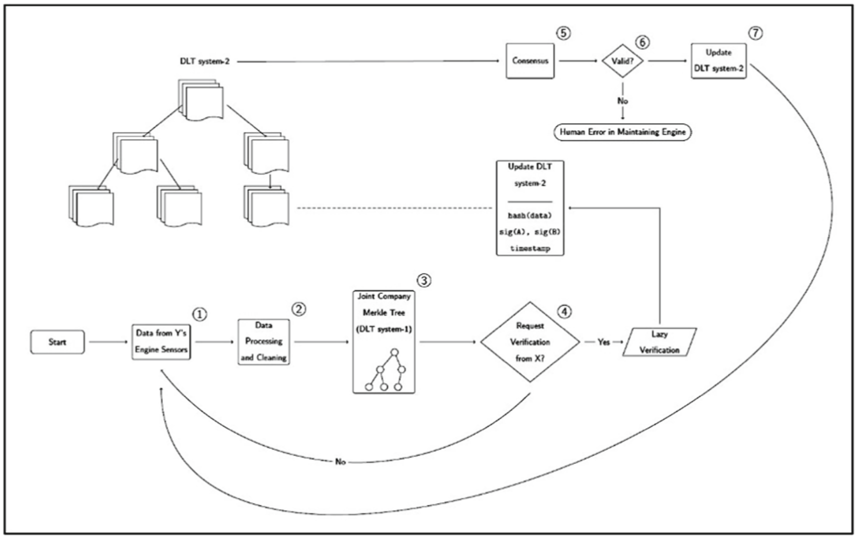
DLT system-2
Consensus
Valid?
Update DLT system-2
No
Human Error in Maintaining Engine
Update DLT system-2
hash(data)
sig(A), sig(B)
timestamp
Start
Data from Y's Engine Sensors
Data Processing and Cleaning
Joint Company Merkle Tree (DLT system-1)
Request Verification from X?
Yes
Lazy Verification
No
①
②
③
④
⑤
⑥
⑦

Fig. 1. Big Data Exchange Use-Case

Figure 1 depicts the process and the related scenario of lazy verification. As shown in the figure, sensor data from Company Y's engine is transported from X's plane to the shared heap, demonstrating Fibereum's ability to function in B2B relationships.

As Fig. 1 shows, in a regular mode of operation: (i) Company Y collects data from the engine. (ii) Y might perform data pre-processing and conditioning. (iii) Y places the data in a verifiable form into DLT System-1. This process (i, ii, and iii) continues as long as there is no valid request for verification, e.g., Company X wishing to terminate the contract with Company Y and replace the engine with a new engine from Company W.) (iv) In case that X, Y, or an authorized third party initiates a legitimate request for verification, X and Y nominate miners (e.g., proxies). (v) The proxies generate local copies of DLT System-1. (vi) A consensus algorithm, such as PoET, is applied and used to construct and/or extend a second DLT system (DLT System-2). For clarity, DLT system-2 is referred to as the global DLT system – that is, a DLT system that has been verified or extended according to a consensus algorithm. If the data is valid, DLT System-2 is updated. Otherwise, an exception is raised. (vii) DLT System-2, the global DLT system, is used to address the exception.

If some blocks do not pass consensus verification due to a malicious act or negligence by one of the parties breaching the agreement, then the matter may be further pursued, e.g., brought to courts or arbitration. Note that in order to save storage space, DLT System-2 might only contain cryptographic pointers to nodes of DLT System-1, thereby serving as a transaction management DLT system.

4.3 Fibereum Permissionless and Permission-Based DLT System Applications

The general mode of operation of Fibereum is permissionless. Nevertheless, in some implementations and use-cases, Fibereum enables permission-based operations via options for access control, compression, and encryption, as well as compression and encryption in tandem [39]. Access control can include password protection access for administrators, system utilities, users, user groups, and the general public. Additional layers of access control can include encryption. Access to utilities for the DLT system construction (e.g., adding blocks) and consensus verification can be subjected to access verification protocols.

The proposed new schemas enable the construction of DLT systems' implementations that are permissionless, permission-based or mixed permission-based/permissionless mode DLT systems. This is enabled via one or more of the following mechanisms: (i) Post verification, some of the generated DLT system blocks (e.g., data, transaction, and contract blocks) might be completely open, i.e., permissionless. Other blocks might be fully protected via access and encryption mechanisms, enabling a mixture of permission-based and permissionless DLT systems. (ii) Parts of plain and encrypted blocks might be available to different entities like a puzzle, where the mechanism to assemble the puzzle pieces is controlled via cryptographic functions. (iii) Multiple copies of the DLT system might increase fault tolerance. Finally, (iv) it should be noted that the level of protection can be related to the sensitivity of the data, where sensitive data might be encrypted and subject to access control.

4.4 Cyber Security and Fault Tolerance Enablement

The proposed Fibereum DLT system schema provides additional security layers for permission-based and permissionless DLT systems. The concept of lazy verification enables storing the data in local storage, where access can be controlled and protected in several ways. The data can be digitally signed, compressed, and encrypted – potentially using methods for tandem compression and encryption. In some implementations, the digests, digital signatures, as well as public and private keys used for encryption can be associated with the physical devices used to generate, transmit, or process the data and with the time that the data was generated. Furthermore, the verification stage can be limited to "trusted parties" and proxies that have protected and potentially permission-based access to the data. Consequently, Fibereum provides better protection against commonly used Blockchain attacks listed in Sect. 3. The following subsection provides further details concerning potential attacks on Fibereum and the resilience of Fibereum implementations to such attacks. This resilience is referred to as "counterattacks."

4.5 Counterattacks by Fibereum

The main threats to Fibereum (and many other DLT systems) are Sybil [20], Majority (51%) attack [3, 21], Denial of Service (DoS) [22, 28], and insider attacks [30]. Several components of the Fibereum DLT system can reduce or completely eliminate the risk emanating from the above and other DLT system attacks. First, when applicable, the option for lazy verification provides ample time to detect and prevent those attacks. Second, some implementations may use the PoET verification protocol for lazy or prompt verification. This increases the resilience of Fibereum to DLT system attacks. Finally, the utilization of permission-based or mixed permission-based and permissionless systems' components can be a paramount counterattack method.

It has been established that without a centralized authority, a system might be susceptible to Sybil attacks [29]. Consensus algorithms (e.g., PoW and PoET) mitigate the effects of Sybil attacks, and permission-based implementations of Fibereum can completely prevent such attacks. Similarly, a Majority Attack would fail with permission-based Fibereum utilizing PoET. In particular, data passing the lazy verification process would enter the second Merkle-based DLT system (System-2), where it would then need to pass through PoET in order for the first and/or the second DLT system to be updated.

Many forms of DoS counterattack methods in general networks exist; some of these methods are applicable to Blockchain DoS [22, 28]. Fibereum can be more resilient to DoS since it is possible that for long periods of time, the only DLT system activity is updating DLT System-1 with new verifiable data, which provides ample time to detect and mitigate a DoS attack.

Insider attacks are the most difficult to prevent, but they often only affect permission-based DLT systems. Measures to reduce the threat of insider attacks have been proposed. Some of these measures are common to many other permission-based systems. Other measures use a blockchain traceability system with a differential traceability algorithm, both of which can be implemented into Fibereum [30].

4.6 Fibereum Smart Contracts

Fibereum Smart contracts are irreversible contracts enforced by the program code embedded in the Fibereum blocks, which are not controlled by users. These contracts are limited by their inability to send HTTP requests and access off-chain data directly. Ethereum can circumvent this limitation via oracles, but this leaves transactions susceptible to attacks that manipulate data and price values [26]. In some implementations, in order to resolve this issue, Fibereum maps the smart contract onto a specific version of an End User License Agreement (EULA) [37]. Hence, the EULA might guide the lazy verification procedure.

5 Additional Fibereum Use-Cases

In this section, several Fibereum use-case examples are presented, concentrating on B2B Applications of the Fibereum DLT system for [Big] Data Exchange use-cases. The concerns related to the data exchange use-cases are data ownership, data rights, data use agreements, managing survivability and termination clauses for contracts, data integrity, liability, monetary value, and responsibility for disclosing the data and its use to third parties, governments, and governing authorities. In these types of use-cases, our objectives include creating a framework for policies governing data exchange in a B2B environment, where data exchange transactions are bounded by a legal contract, potentially in the form of license agreements or subscriptions (signed or click-through), specifying certain terms, such as ownership, usage policies, rights, management, and governance. Often, these agreements take the form of End User License Agreements, Developer License Agreements, and Data License Agreements. Given a predetermined legal contract, Fibereum aims to minimize the computational burden of consensus verification. It should be noted that Fibereum also provides efficient mechanisms for support of other use-cases, including digital currency exchange, B2C data exchange, as well as services related to data exchange. This section includes examples of these use-cases as well.

Some of the use-cases might include additional Fibereum-based DLT systems, e.g., a DLT system for transaction management that records the process of data exchange and a smart contract DLT system that is used to dictate data usage, rights, and termination. The use-case examples, however, do not elaborate on the internals of the smart contract DLT systems and their operation. Finally, all the use cases may deploy a permissionless version of Fibereum, a permission-based version, or a combination.

5.1 Exception Maintenance 2: Data Networks

In some Fibereum implementations, DLT System-2 is a Blockchain DLT. As an example of such an implementation, one can consider a use-case where company U is a process control firm that has sensors installed in an oil refinery that belongs to company V. In this implementation of the Fibereum DLT system, the data is stored locally by the data stakeholders (e.g., by companies U, V, and their affiliates/proxies) in Merkle heaps. Additionally, a Fibereum DLT system for transaction management may record the process of data exchange, and a Fibereum smart contract DLT system may be employed to dictate data usage, rights, and termination. When new sets of sensor data are available, they are aggregated into blocks, and the blocks are inserted into the Merkle heaps. At the same time, the transaction management DLT system is updated. At the time that consensus verification is mandated (e.g., a dispute between Company X and Company Z or a discovery subpoena by local authorities due to an accident), the integrity of the data stored in the heaps and proxies is assessed.

The following is a flowchart of the of DLT system-2 construction and lazy verification procedure applied in the Exception Handling use cases:

The Lazy Verification Procedure

```
Require: LazyVerification(DLT system-1, DLT system-2)
 1: while DLT system-1 is non-empty do
 2: d ← POP(DLT system-1)
 3: if d is valid then
 4:     APPEND(DLT system-2, d)
 5: else if d is invalid then
 6:     raise legal issue
 7: end if
 8:  end while
```

5.2 Digital Currency

This use-case considers digital currency applications that are similar to Bitcoin and Ethereum digital coins exchange. In contrast to most other digital coin DLT systems, both the intra-block and the inter-block may be managed via Merkle trees. Due to the nature of the application and potential attacks, the verification may be prompt and incentivized using fees or digital coin mining rewards. A PoET consensus mechanism may be employed to reduce operational complexity and energy consumption and improve counterattack capabilities.

5.3 Targeted Advertising

This use-case may be a B2B or a B2C use-case. For example, assume that Company P manufactures autonomous vehicles and Company Q, or a consumer R, uses these vehicles, which collect federated data along with sensor data. Specifically, suppose that a consumer X buys a car manufactured by Company Y, and Company Z wishes to access parts of the sensor data from the car. The process is similar to the process described in the above exception maintenance use-cases. However, it might utilize two Merkle heaps or one heap with access control to heap elements. Both classified and unclassified information is accessible to X and Y via one heap, but, for the protection of X's privacy, the second heap contains only unclassified information for Z.

5.4 Patient Medical History

This use-case considers situations where patient X wishes to switch from care provider Y to care provider Z and then transfer their medical history to Z. Due to stringent confidentiality requirements, it is most likely that the DLT system implementation would be permission-based, preventing unauthorized access to medical records. Any new medical data that goes into the records by the care provider must be verifiable and potentially include encryption, digests, and the digital signature of the patient before it is added as a block to the DLT. Provider Y uses its own encryption, digests, and digital signature to securely access medical records. If the patient wishes to share their information with other care providers, e.g., Z, then Provider Y might require a digital signature of Patient X and Provider Z for consent to release information. The DLT system includes a network of care providers, as medical records may need to be transferred from one care provider to another care provider. In this case, the medical records and other information are encrypted in a Merkle heap (DLT system-1) and, following verification, sent to other care providers through the Merkle tree of DLT System-2. Lazy verification, initiated on a "need to do" basis, e.g., switching a care provider, is used to check permissions and verify information correctness.

5.5 Digital Cartography

This use-case considers a situation where the system includes satellites, e.g., X1, X2, X3, and X4, a ground station Y, a user Z, and an object of interest W. The information generated by the remote sensing satellites and gathered by the ground station (e.g., GPS locations of Object W) is stored in the Merkle heap-based DLT System-1 and is accessible to User Z. The system allows User Z to request a legitimate verification of certain parts of the information. This triggers lazy verification and the creation of DLT System-2. The verification may include proxies. Since storage is placed within a heap, the location may be constantly updated, and User Z can constantly update the positions of Object W by requesting lazy verification. If an exception occurs, the data stored in the heap can be used to track previous locations with precise timestamps from the satellites' atomic clock in order to help figure out what may have happened.

5.6 Non-fungible Tokens

This use-case considers situations where a DLT system is used for the exchange and management of Non-Fungible Tokens (NFT) [10]. In contrast to most other NFT DLT systems, both the intra-block and the inter-block may be managed via Merkle trees. Due to the nature of this application and potential attacks, the verification may be prompt and incentivized using fees[1].

5.7 Smart Contracts and Licensing Agreements

This pertains to cases where a DLT system is used for smart contract management. In contrast to most other smart contract DLT systems (e.g., Ethereum-based smart contracts), both the intra-block and the inter-block may be managed via Merkle trees. Due to the nature of the application and potential attacks, the verification may be prompt and incentivized using fees[1].

USCG is interested in exploring, along with our team, the utility of DLT systems for these use-cases and may utilize the DLT systems for licensing, e.g., licensing of fishing companies and vessels. Given that the DLT users are not necessarily USCG staff members, the system may have to tighten security measures with respect to access to the DLT and the construction of DLT system blocks.

5.8 Copyrighted Material

This use-case considers situations where Consumer X wishes to access copyrighted material produced by Company Y. In a possible DLT system implementation, the operation procedures are similar to the operating procedures of the DLT system described in Use-case 4.1. However, the amount of data stored in DLT System-1 is not as big as the amount of data expected in Use-case 4.1. Furthermore, the DLT system may be permission-based so that only Consumer X and Company Y can access the material. Release of the material requires the consent of both parties, i.e., Consumer X and Company Y must both sign in order to sell content to a third party. Note that this use case has some overlap with NFT and can be used as a DLT system for NFT.

5.9 Commerce, Supply Chain Management, and Inventory Management Systems

This use-case considers situations where Company X wishes to transport goods to a warehouse owned by Company Y. In this case, a comprehensive database accessible by both X and Y can be used. This DLT system is likely to be permission-based so that only the two parties and their affiliates have access to it. The original owner of the DLT system, Company X, shares parts of the database of verifiable transactions with Company Y in a Merkle heap-based DLT System-1. Lazy verification may be used to generate DLT System-2 in order to mine data, verify transactions, manage inventory, and validate the integrity of the data and the underlining supply chain. Inventory management can be implemented in a similar way.

The US Coast Guard (USCG) is interested in exploring, along with our team, the utility of DLT systems for these use-cases and may utilize DLT systems under the assumption that the users are internal to the organization. Hence, they may have "some" level of trust by the system (e.g., after supplying credentials that associate them with the USCG).

5.10 Weather Broadcasting

This use-case considers a situation where two or more weather stations (e.g., X and Y) wish to share data regarding weather conditions in a certain region. The DLT system may be permission-based so that only Stations X and Y can access the data. Sensor data created by X and Y is verifiable and stored in DLT System-1. Lazy verification and the

[1] A PoET consensus mechanism may be employed to reduce operation complexity and energy consumption and improve counterattack capabilities.

creation/update of DLT System-2 may take place when there are discrepancies between X and Y as to the data relied upon or in their weather prediction.

6 Conclusion and Future Research

The DLT systems and methods discussed above include novel modifications to blockchain technology, providing a superior framework for DLT systems. Those modifications can improve the cost/performance of DLT systems in current applications and extend the utility of current DLT systems to several new applications and use-cases. The Fibereum DLT system has various applications in the fields of Big Data, including Transportation, Smart Cities, Healthcare, Process Control, and Internet of Things. Fibereum DLT systems are also suitable for other applications, such as Digital Currency, Smart Contracts and licensing, and Supply Chain Management.

Future work can include: (i) Implementations for other use-cases and data exchange applications; (ii) Monetization and control of federated data; (iii) Enhancements to B2C applications where concerns may include tight privacy constraints, as well as consumer rights protection; (iv) Further exploration of the utility of additional cryptographic data structures and other non-linear data structures, e.g., directed acyclic graphs for transactions and/or data storage; (v) Appending new data transaction information to a concurrent transaction data structure rather than directly appending it to the DLT; (vi) Appending new sensor data to a concurrent sensor data structure rather than directly appending it to the DLT; (vii) Further exploring the management of federated data where different parts of copies of the data reside in the DLT systems of individual parties; (viii) Exploring implementations where the data is compressed and encrypted, potentially for enabling permission-based access.

Acknowledgment. This material is based in part upon work supported by the Department of Homeland Security grants TXST83938 and E2055778 and by the National Science Foundation under Grant CNS-2018611 and CNS-1920182.

References

1. Narayanan, A.: Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction. Princeton University Press, Princeton (2016)
2. Lipton, A., Treccani, A.: Blockchain and Distributed Ledgers: Mathematics, Technology, and Economics. World Scientific, Singapore (2022)
3. Nakamoto, S.: Bitcoin: A Peer-to-Peer Electronic Cash System. http://www.bitcoin.org/bitcoin.pdf
4. Wood, G.: Ethereum: a secure decentralised generalised transaction ledger. Ethereum Proj. Yellow Pap. **151**, 1–32 (2014)
5. Horowitz, E., Sahni, S.: Fundamentals of Data Structures. Sung Kung Computer Book Co. (1987)
6. Androulaki, E., et al.: Hyperledger fabric. In: Proceedings of the Thirteenth EuroSys Conference. ACM (2018)
7. Introduction to smart contracts. https://ethereum.org/en/developers/docs/

8. Muller, M.: Essentials of Inventory Management. HarperCollins Leadership, Nashville (2019)
9. Chopra, S., Meindl, P.: Supply chain management. Strategy, planning & operation. In: What's New in Operations Management. Pearson (2018)
10. Fraga-Lamas, P., Fernandez-Carames, T.M. Leveraging distributed ledger technologies and blockchain to combat fake news (2019). CoRRabs/1904.05386. arXiv:1904.05386. http://arxiv.org/abs/1904.05386
11. Wang, Q., et al.: Non-Fungible Token (NFT): Overview, Evaluation, Opportunities and Challenges (2021). arXiv:2105.07447
12. Merkle, R.: A certified digital signature. In: Crypto, vol. 89, pp. 218–238 (1989)
13. Castro, M., Liskov, B.: Practical byzantine fault tolerance and proactive recovery. ACM Trans. Comput. Syst. **20**(4), 398–461 (2002)
14. Dabbagh, M., et al.: A survey of empirical performance evaluation of per-missioned blockchain platforms. Comput. Secur. **100**, 102078 (2021)
15. Pal, A., Kant, K.: DC-PoET: proof-of-elapsed-time consensus with distributed coordination for blockchain networks. In: 2021 IFIP Networking Conference (IFIP Networking), pp. 1–9 (2021)
16. Peng, K.: A general, flexible, and efficient proof of inclusion and exclusion (1970)
17. Proof-of-stake (POS). https://ethereum.org/en/developers/docs
18. Liu, Y.C.: Performance of a CSMA/CD protocol for Local Area Networks. https://ieeexplore.ieee.org/document/1146621
19. Tenenbaum, A.S.: Computer Networks. Prentice Hall, Indore (2009)
20. Heilman, E., Kendler, A., Zohar, A., Goldberg, S.: Eclipse attacks on bitcoin's peer-to-peer network. In: usenixsecurity15 (2015)
21. Iqbal, M., Matulevi˘cius, R.: Exploring sybil and double-spending risks in blockchain systems. IEEE Access **9**, 76153–76177 (2021)
22. Joshi, J., Mathew, R.: A survey on attacks of bitcoin. In: Proceeding of the International Conference on Computer Networks, Big Data and IoT (ICCBI - 2018), pp. 953–959 (2020)
23. Frankenfield, J.: 51% attack: definition, who is at risk, example, and cost (2022). https://www.investopedia.com/
24. Raikwar, M., Gligoroski, D.: DoS Attacks on Blockchain Ecosystem (2022)
25. Jansen, M.: Do smart contract languages need to be turing complete?" In: Blockchain and Applications, pp. 19–26 (2020)
26. Ethereum smart contracts. In: 2020 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE). IEEE (2020)
27. Torres, C.F., Camino, R., State, R.: An empirical study of frontrunning on the ethereum blockchain. In: Usenixsecurity21 (2021)
28. Ali Khan, Z., Siami Namin, A.: A Survey on Vulnerabilities of Ethereum Smart Contracts (2020). ArXiv 2012.1448
29. Chen, H., et al.: A Survey on Ethereum Systems Security: Vulnerabilities, Attacks and Defenses (2019). ArXiv.1908.04507
30. Samreen, N.F., Alalfi, M.H.: SmartScan: an approach to detecting denial of service vulnerability in ethereum smart contracts. CoRR abs/2105.02852 (2021). arXiv: 2105–02852. https://arxiv.org/abs/2105.02852
31. Gojka, E.E., et al.: Security in distributed ledger technology: an analysis of vulnerabilities and attack vectors. In: Intelligent Computing, pp. 722–742 (2021)
32. Putz, B., Pernul, G.: Trust factors and insider threats in permissioned distributed ledgers - an analytical study and evaluation of popular DLT Frameworks. https://core.ac.uk/outputs/232204350
33. Davenport, A., Shetty, S., Liang, X.: Attack surface analysis of permissioned blockchain platforms for smart cities. In: 2018 IEEE International Smart Cities Conference (ISC2), pp. 1–6 (2018)

34. Cäsar, F.: Cerberus: A parallelized BFT consensus protocol for radix (1970). https://api.semanticscholar.org/CorpusID:221297416
35. Snow, P., et al.: Factom Ledger by Consensus (2015). https://cryptochainuni.com/wp-content/uploads/Factom-Ledger-by-Consensus
36. Kaur, G., Gandhi, C.: Chapter 15 - scalability in blockchain: challenges and solutions. In: Handbook of Research on Blockchain Technology, pp. 373–406 (2020)
37. Gai, K., Wu, Y., Zhu, L.Q., Shen, M.: Privacy-preserving energy trading using consortium blockchain in smart grid. IEEE Trans. Industr. Inf. **15**(6), 3548–3558 (2019)
38. Zhu, L. Wu, Y. Gai, K., Kwang, K., Choo. R.: Controllable and trustworthy blockchain-based cloud data management. Future Gener. Comput. Syst. **91**, 527–535 (2019)
39. Cotton, H., Bolan, C.: User perceptions of end user license agreements in the smartphone environment. In: Australian Information Security Management Conference, pp 235–244 (2018)
40. Bubel, R., Hähnle, R., Geilmann, U.: A formalisation of java strings for program specification and verification. In: Software Engineering and Formal Methods
41. Tamir, D., Bruck, D.: Compression and Decompression Engines and Compressed Domain Processors, US Patent 10404277 (2019)