Preprint
Kuhn, R, Yaga, D, Voas, J. Rethinking distributed ledger technology. *IEEE Computer*, *52*(2), 68-72. (2019)

Rethinking Distributed Ledger Technology

Rick Kuhn, Jeff Voas, Dylan Yaga
National Institute of Standards and Technology

Distributed ledger technology (DLT) offers new and unique advantages for information systems, but some of its features are not a good fit for many applications. We review the properties of DLT and show how two recently developed ideas can be used to retain its advantages while simplifying design.

Introduction

While most of the excitement around blockchain stems from its use in cryptocurrencies, designers are beginning to find interesting ways to solve system problems using blockchain and other forms of distributed ledger technology (DLT). The most commonly used data structure for distributed ledgers is the blockchain. A key feature of a blockchain based system is the decentralized, replicated data synchronized among separate network nodes, which may be geographically dispersed. There is substantial discussion around some terms in DLT, in particular public vs. private systems. We believe it is better to distinguish blockchain systems based on their permission model: permissioned or permissionless; since it is directly tied to the technology; where private or public may apply to the visibility of the network or ledger itself.

With its features providing distributed, trusted data using no central server, DLT seems a natural tool for many complex distributed systems, and a number of implementations have been proposed. However, some environments and applications are not well suited to using an append-only ledger. For example, an analysis of DLT for the international banking consortium SWIFT found that the permissionless model used by Bitcoin and other cryptocurrencies "does not provide the level of trust, transparency, and accountability required by the financial industry" [1]. The SWIFT analysis noted that permissioned ledgers are helpful, but "existing implementations of permissioned ledgers remain basic". Of particular concern is the "immutable" aspect of transactions recorded in blockchains [2][3] [4]. As noted by the European Banking Institute, "once an error is embedded in the blockchain, this may be highly problematic, legally, in that often law requires the ability to rectify errors as a matter of law in a way foreign to DLT" [2]. One option is to correct errors by issuing a new transaction which supersedes the older erroneous transaction. In this way, the ledger provides a full history of events as they happened. While this is possible or desirable for some applications, privacy laws lead to additional complications, as discussed later.

In addition to complicating support for privacy rules, other properties of conventional blockchains are not a good match for applications beyond cryptocurrency [5] [6], and modifications to distributed ledger designs are being developed to meet new needs. Blockchains are a valuable distributed ledger technology for providing trust, but there are

many ways to construct distributed ledgers. We propose an alternative that provides trust features of blockchains with a more flexible data structure and ordering protocol.

DLT and Data Management

A distributed ledger, as the name suggests, is a distributed record of transactions, maintained by consensus among a network of peer to peer nodes (possibly geographically dispersed). The most widely recognized form of DLT is the blockchain structure, which provides the basis for cryptocurrencies and a variety of other applications. Most currently available distributed ledger designs using blockchain provide certain properties:

- *Pseudo-anonymity* – especially for cryptocurrency, blockchains enable participation using only identifiers. Permissioned blockchains may not include this property.
- *Public access, transparency* – every participant can see all transactions on the blockchain, although they may be anonymized. This property may also not be provided in permissioned systems.
- *Small transaction size* – Blockchains were originally designed for monetary transactions, so messages are assumed to be relatively small.
- *Immutable records* – As a consequence of the linked chain of cryptographic hashes of records, a change to one record would cause the hash of subsequent records to be invalid, so changes require recomputing the entire chain. As a result, it is generally intractable to change any record in a blockchain.
- *Proof of work or other expensive consensus models* – A consequence of the need to prevent double spending. Permissioned blockchains do not generally need this feature and can use simpler consensus.
- *Block ordering guarantee* – the consensus mechanism ensures ordering of the blocks and therefore transactions, preventing the possibility of double spending.
- *Decentralization* – there is no central authority for records. With each update, records are dispersed to peer nodes simultaneously, who ensure the updates are correct.
- *Replication and Synchronization guarantee* – transactions are duplicated across all nodes of the network, so that every node has an identical copy of all transaction records, current to the most recent update cycle. Consensus protocols are designed such that when the consensus is complete, all nodes have an identical copy of the distributed ledger records.
- *Integrity protection* – Cryptographic hashes are used to guarantee that records have not been changed.

We compare these properties with the needs of more typical applications of distributed data storage and retrieval in Table 1. Note that six of the nine blockchain properties designed for cryptocurrency are at odds with the requirements of many other

applications.

Cryptocurrency	**Finance, supply chain, e-commerce, etc.**
1. Pseudo-Anonymity	ID required for contracts or government regulation
2. Public access, transparency	Controlled access
3. Small transaction size	Range of message sizes up to large documents, images
4. Immutable records	Changes and deletions, often required by law
5. Proof of work and other expensive consensus models	Flexible consensus models
6. Block ordering guarantee	Timestamps often required
7. Decentralization	Same in many applications
8. Replication and Synchronization guarantee	Same in many applications
Integrity protection	Same in many applications

Table 1. Comparing characteristics of DLT applications

Need for an Alternative Solution

The mismatch between blockchain properties and many application needs has led to a number of problems in applying blockchain designs to data management problems. For example, Bitcoin is designed to provide some degree of anonymity in transactions (i.e., only public identifiers, not real-world identities are used), but the law may prohibit anonymity for many types of transactions and require participants to be identified for tax or other purposes. Laws such as the European Union General Data Protection Regulation (GDPR), that require the ability to delete privacy relevant information, may limit the type of information that can be stored in a blockchain [2] [4].

For system engineers, the price of distributed trust is often added complexity. The design choices that were made to incorporate anonymity and prevent double spending in blockchains often lead to seemingly unnecessary complications when applied to areas beyond cryptocurrency. For example, immutability has resulted in designs where alterable records must be kept off of the blockchain, with only pointers to them stored in the blockchain itself. Alternatively, some designs involve encrypting data on the blockchain, then destroying the encryption key to "delete" the data. Neither of these options may be desirable for many applications, as the first option leads to unnecessary complications, and the second risks the data being decrypted in the future, when data must be protected for decades. These are serious design issues for supporting privacy requirements such as those of GDPR, resulting in proposals such as an "editable blockchain" [7] using new forms of hashing. For cryptocurrency, a consensus algorithm is needed to guarantee record ordering in the absence of a central time authority (i.e., transactions are ordered based on group consensus, rather than time of entry into a system), and this ordering is used to prevent double spending. Designs for access control using blockchain may involve tokenizing permissions, then passing these to users, and spending down the value to remove a permission from a user. All of these strategies are needed to take advantage of blockchain's trust properties, but blockchains would probably not be used if a more conventional database could provide the desired distributed trust.

At first glance, blockchain solutions for applications such as supply chain, financial settlement, and others may appear to offer nothing more than added complexity in comparison with a conventional database. However, when more than one organization is involved, the decentralized trust of blockchains and other distributed ledgers can be a tremendous advantage. For example, consider regulated industries where audit is a part of doing business. Every node on the system can have a full set of records detailing the movement of assets. Any shared database can keep track of asset movement, but DLT adds trust by maintaining current, integrity-protected records at every organization, making it easy to audit the process. Thus, the financial industry views *full traceability* and *simplified reconciliation* of transactions among the key advantages of DLT [1]. We can view DLT as adding a layer of distributed trust to the problem of data storage and retrieval, clearly a desirable property, but industry is still struggling with how to use DLT in practical ways.

A Permissioned Distributed Ledger Model for Decentralized Trust

Much of the current DLT research seems to center on how to get around properties that were baked into blockchain. Adaptations such as faster consensus algorithms are gradually moving DLT from its origin in cryptocurrency towards a more general-purpose database technology. But instead of tweaking blockchain designs, we can rethink the idea of a distributed ledger to reflect the needs of data management applications as discussed earlier.

Can we provide a simpler model, that gives the decentralized trust of a blockchain, but otherwise behaves as a conventional database? In this section, we describe an approach to achieving this goal, using two recent proposals: a *data block matrix* [8], and *verified time* [9]. The data block matrix retains hash-based data integrity guarantees, while allowing controlled modification or deletion of specified records, with integrity guarantees for all other records. A data block matrix can be implemented in a decentralized system to provide data replication among peers. The verified time protocol allows guaranteed timestamps to be used in place of consensus algorithms to ensure record ordering.

The data block matrix uses an array of blocks, with hash values for each row and column. This structure makes it possible to delete or modify a particular block with hash values assuring that other blocks have not been affected. An example is shown in Fig 1. Suppose that it is desired to delete block 12, by writing all zeroes to that block, or otherwise modifying it. This change disrupts the hash values of $H_{3,-}$ and $H_{-,2}$ for row 3 and column 2. However, the integrity of all blocks except the one containing "X" is still ensured by the other hash values. That is, other blocks of row 3 are included in the hashes for columns 0, 1, 3, and 4. Similarly, other blocks of column 2 are included in the hashes for rows 0, 1, 2, and 4. Thus the integrity of blocks that have not been deleted is assured.

Blocks can be deleted by overwriting with zeroes or other values, with one row and one column hash recalculated; specifically, after deleting block *i*, *j*, row *i* and column *j* hash values are recalculated.

As shown in [8], the data structure ensures the following properties:

- *Balance*: upper half (above diagonal) contains at most one additional cell more than the lower half.
- *Hash sequence length*: number of blocks in a row or column hash proportional to $\sqrt{N}$ for a matrix with *N* blocks, by the balance property.
- *Number of blocks*: The total number of data blocks in the matrix is $N^2 - N$ since the diagonal is null.
- *Block dispersal*: No consecutive blocks appear in the same row or column

	0	1	2	3	4	
0	•	1	3	7	13	$H_{0,-}$
1	2	•	5	9	15	$H_{1,-}$
2	4	6	•	11	17	$H_{2,-}$
3	8	10	12	•	19	$H_{3,-}$
4	14	16	18	20	•	$H_{4,-}$
	$H_{-,0}$	$H_{-,1}$	$H_{-,2}$	$H_{-,3}$	$H_{-,4}$	etc.

Figure 1. Data block matrix with numbered cells

Clearly, this data structure is not suited to all DLT applications, but it offers features that are difficult to provide with a conventional blockchain. Our goal is not to replace blockchains, but to offer a new form of data storage structure that provides the integrity guarantees of blockchain with the addition of reversibility, which can be used in a wide range of applications. A comparison is shown in Table 2.

Blockchain – provides integrity, sequencing	**Data block matrix** – provides integrity, erasure
Integrity protection, no erasure possible	Integrity protection for all blocks not erased
Double-spend problem solved by distributed transaction ordering guarantees	Ability to erase values obviates need for ordering guarantees through consensus algorithms
Ordering guarantees require consensus algorithms	Ordering guarantees granted by time authority

Table 2. Blockchain and data block matrix features

In distributed ledger designs, the role of time is often an afterthought. Some DLT systems have no inherit transaction timestamp, to record when the transaction was submitted to the system. Rather the transactions adopt the time in which they were included into the ledger which may occur after a significant amount of time has passed since being submitted. This approach has worked for applications where just having a transaction accepted is good enough (e.g., we do not need to know that a cryptocurrency transaction was submitted down to the millisecond, just that it was indeed submitted and eventually recorded in the ledger).

However, when time dependent situations arise, a timestamp becomes more

important, and knowing when a transaction was submitted to a system may be more important than when it was incorporated into the ledger. Often, systems will rely on local system time, or a network time – both may differ from one system to the next. Distributed ledgers must be able to operate in environments that include rules mandated by government or contract. For some applications, the ordering of transactions into blocks within the blockchain may not be enough, and there is a need for a global timestamp service, providing verified time [9]. Time is a key component of this because things happen outside of the blockchain that matter for applications – orders must be fulfilled by a specified date and time, legal papers filed on schedule, and so on, requiring timestamps of events that take place outside of the blockchain.

A global time-stamping approach would include an agreed upon and accepted service. This approach could incorporate a high resolution blockchain time mechanism, such as the open-source Chainpoint protocol [10], into the distributed ledger to produce a final and agreed upon timestamp. Chainpoint uses the Network Time Protocol with the NIST Randomness Beacon to provide provable timestamps and might be adapted to the needs described here.

Conclusions

The blockchain data structure and proof-of-work protocol were designed to solve the problem of double spending in cryptocurrencies. Although blockchain has found many applications outside of cryptocurrency, many of its features are not well suited to common data management applications, leading many to argue that distributed ledgers are only databases with more complex features. As we have described, the added trust of distributed ledgers is a valuable feature, providing greatly simplified auditability and verification of actions among multiple parties in applications such as supply chain and others.

The blockchain design for hash-based integrity verification provides trust, at the cost of an inability to delete or update records, leading to design complications that would not arise with conventional database management systems. Similarly, the sequencing guarantees of blockchain consensus protocols are needed for cryptocurrency in the absence of a universal timestamp. Moreover, actions within the distributed ledger must be connected with other actions in the real world, through accurate timestamps. We have presented a new architecture that provides the trust features of blockchains, with characteristics that allow for simpler designs and greater practicality in conventional data management problems. We believe this alternative can lead to new approaches to incorporating trust into distributed systems applications.

References

[1] Le Borne, F., Treat, D., Dimidschstein, F., & Brodersen, C. (2017). SWIFT on distributed ledger technologies-Delivering an industry-standard platform through community collaboration.

[2] Zetzsche, D.A., Buckley, R.P. and Arner, D.W., 2018. The distributed liability of distributed ledgers: Legal risks of blockchain. U. Ill. L. Rev., p.1361.

[3] M.Berberich, M.Steiner,"Blockchain Technology and the GDPR," 2 Eur. Data Protection Law Review. 422, 2016.
[4] O. Kharif, "Is Your Blockchain Doomed?", Bloomberg Business Week, Mar. 22, 2018.
[5] T. Simonite, "Banks Embrace Blockchain's Heart but not it's Soul", MIT Technology Review, Sept. 24, 2015.
[6] UK Govt. Chief Scientific Advoser, "Distributed Ledger Technology: beyond Block Chain", https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/492972/gs-16-1-distributed-ledger-technology.pdf
[7] J. Condliff, "Is an Editable Blockchain the Future of Finance?", MIT Technology Review, Sept. 20, 2016.
[8] D.R. Kuhn, A Data Structure for Integrity Protection with Erasure Capability, NIST Cybersecurity Whitepaper, May 31, 2018.
[9] Stavrou, A., & Voas, J. (2017). Verified Time. Computer, 50(3), 78-82.
[10] Chainpoint – Innovations in Blockchain Timestamp Proofs. https://blog.tierion.com/chainpoint-innovations-in-blockchain-timestamp-proofs