brillio

Building Enterprise solutions with AWS Managed Blockchain and QLDB

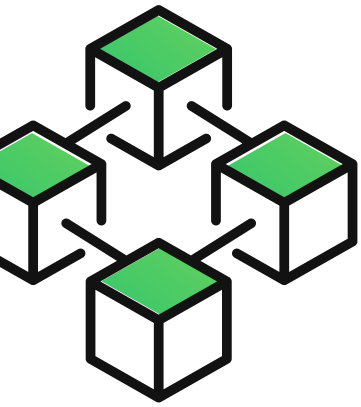
brillio

Foreword

Blockchain technology is becoming increasingly popular, providing "an open, distributed ledger that can record transactions between two parties efficiently and in a verifiable and permanent way."

Today, building a scalable blockchain network with existing technologies is complex to set up and difficult to manage. To create a blockchain network, each network member needs to manually provision hardware, install software, create, and manage certificates for access control and configure networking components. Once the blockchain network is running, you need to continuously monitor the infrastructure and adapt to changes, such as an increase in transaction requests or new members joining or leaving the network.

To remove the complexity associated with blockchain networks, AWS announced Amazon Managed Blockchain, a fully managed service that makes it easy to create and manage scalable blockchain networks.



AWS also announced Amazon Quantum Ledger Database (Amazon QLDB), a ledger database that provides some of the same features as blockchain for data integrity. It is designed for centralized systems where there is a central trusted authority. Amazon QLDB provides a transparent, immutable, and cryptographically verifiable transaction log owned by a central trusted authority.

brillio

Introduction

Let’s deep dive into some of these concepts, the challenges people face building with blockchain, and how the AWS Managed blockchain and QLDB solve these issues.

First, let's take a look at some of the technology behind blockchain: Ledgers, Decentralized networks, Consensus algorithms, and Smart contracts.

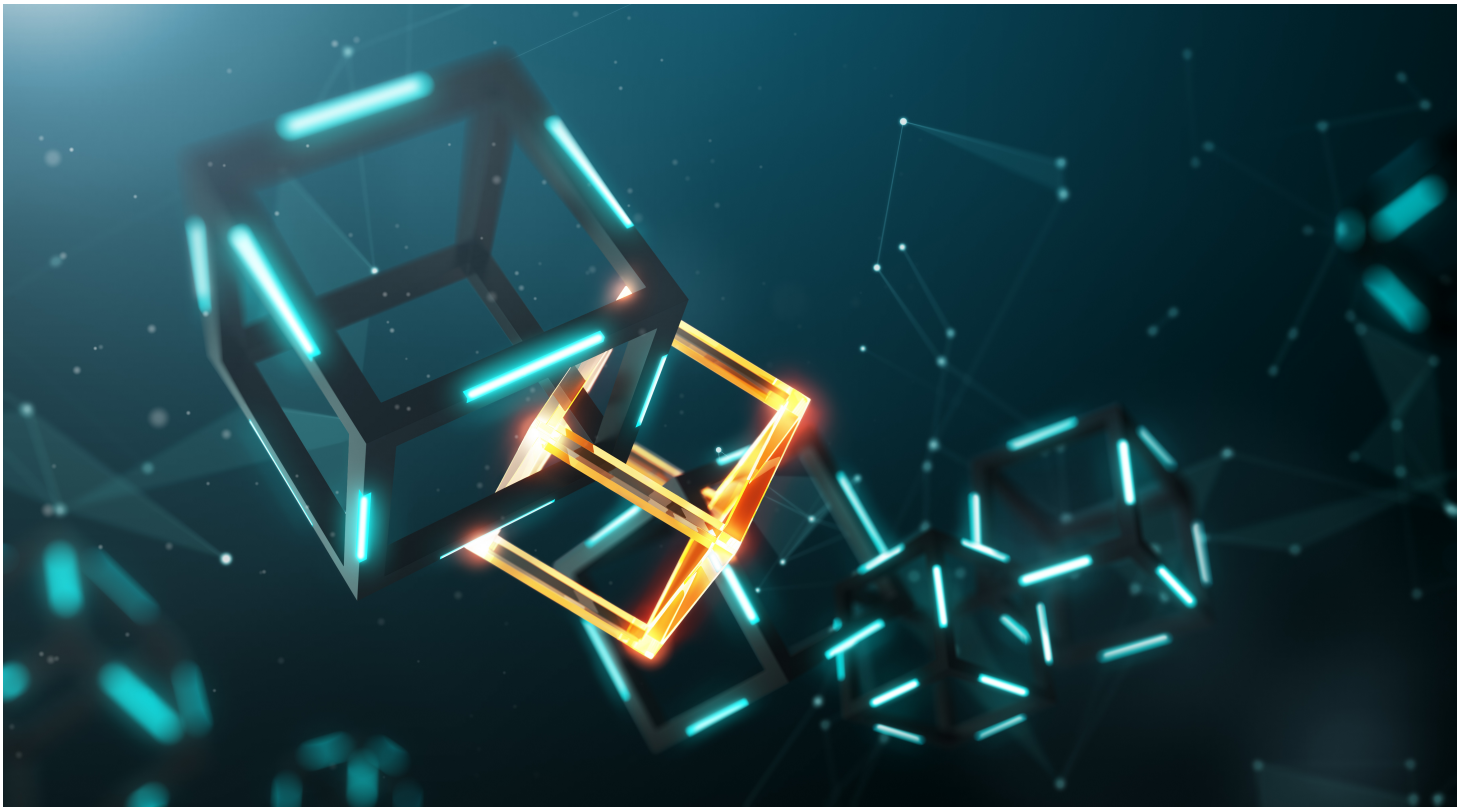
Blockchains are designed to maintain the integrity of data. They are immutable—committed data history cannot be altered or deleted, it can only be updated. Ledgers are the building blocks that help make blockchains immutable, so let’s look at this concept in detail.

Ledgers have been around for a long time and were typically used to record a history of economic and financial activity between two or more parties. Today, a banking application that tracks credits and debits is one of the most common examples of a ledger.

Ledgers found in a blockchain typically consist of the following:

- Current and Historical state: A data structure that keeps the current and historical state values, allowing applications to easily access the data without needing to traverse the entire transactional log.

- A journal: A transactional log that keeps a complete record of the entire history of data changes. The transactional log is append-only, meaning that each new record is chained to the previous, allowing you to see the entire lineage of the data's change history. Additionally, with the help of cryptographic hashing, a process that assigns a unique identifier to each record, blocks are chained to one another. This allows ledgers to have a timekeeping property allowing anyone to look back in time and get proof that the data transaction occurred, making auditing simple.



Compare this to relational databases where customers must engineer an auditing mechanism because the database is not inherently immutable. Such auditing mechanisms built with relational databases can be hard to scale. They put the onus on the application developer to ensure that all the right data is being recorded.

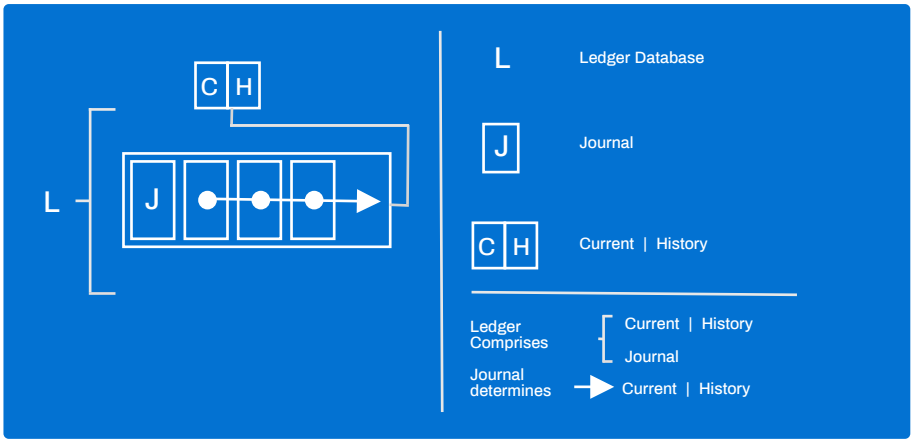


Fig1. Ledgers in Blockchain

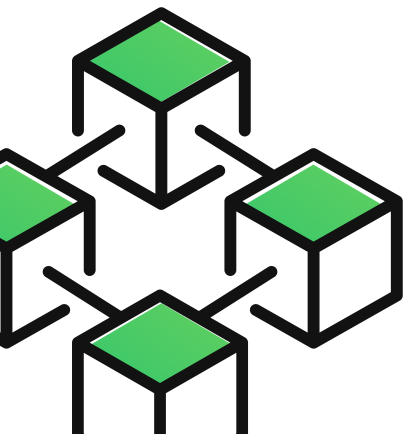
In addition to a distributed ledger, blockchains also include a consensus mechanism and smart contract execution environment.

- Consensus algorithms help ensure that the members in the network have an agreement method to allow transactions and data to be committed to the ledger. If the consensus requirements are not met, then the transaction is considered not valid.

- Smart contracts are programs that have the rules and penalties of engagement for a contract defined into lines of code. The program continuously checks when the conditions for a contract are met and then ensures that the contract gets automatically executed.

Together, these elements allow two or more parties to transact with decentralized trust, where each party consents to the transaction and records the transaction. Decentralized trust makes sense when

multiple organizations must independently verify transaction history and have a single, up-to-date, accurate view of data. It also makes sense when there is no single party that wants to maintain an application, but network members still want to transact with other parties efficiently



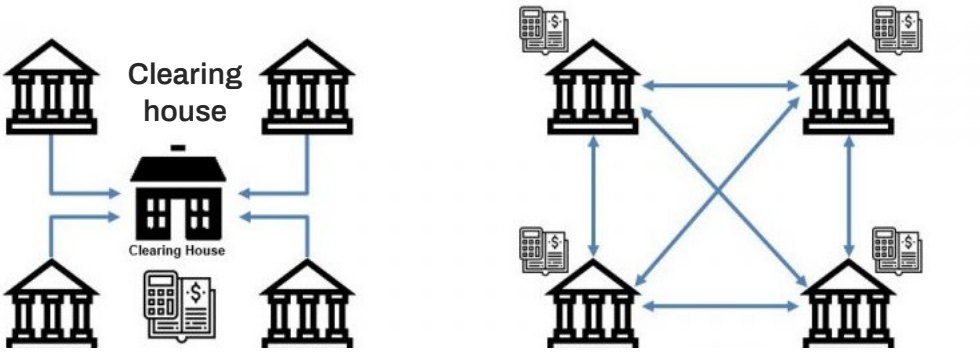
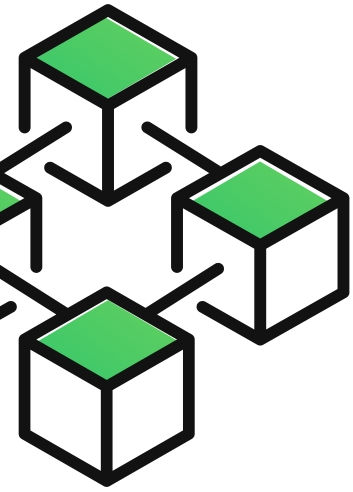


Fig 2 Ledger Types

Let us take the example of a trade finance application where decentralized trust is needed. Trading goods across international borders requires many organizations to work with one another, such as exporters, importers, shipping companies, multiple banks, insurance companies, and customs departments. With the number of parties involved there is no trusted central authority. Each party wants to independently verify the documentation related to the trade and doesn't want any single entity to own the record of activity.

The current process requires trade-related paperwork to go back and forth between the parties, which can take time (5-10 days) to complete. This results in long processing time and high costs.



In this scenario, enabling the parties to operate with decentralized trust improves efficiency and cuts down costs. A single participant does not own the infrastructure and the system distributes a copy of the transaction ledger to each participant for independent verification. The business contract, such as a letter-of-credit, can be written as a smart contract in the blockchain application. It can automatically execute as soon as all parties provide consensus to record the transaction.

Customers look to blockchain as technology that enables them to transact with multiple parties when there is no single trusted authority and they need a system with decentralized trust.

Managed Blockchain –

Addressing customer's challenges to maintain blockchain network

It is difficult, expensive, and time-consuming to create and manage blockchain networks using existing frameworks. First, to create a blockchain network with permissions each network member must manually provision hardware, install software, create and manage certificates for access control, and configure networking components. When the blockchain network is running, users must continuously

monitor the infrastructure. They must adapt to changes, such as an increase in transaction requests or new members joining or leaving the network.

To help overcome the barriers that people face trying to build using blockchain, AWS has created Managed Blockchain. Unlike a self-hosted blockchain network, Amazon Managed Blockchain eliminates the need for manually provisioning hardware, configuring software, and setting up networking and security components. This service allows users to set up and manage a scalable blockchain network with just a few clicks. It automatically scales to meet the demands of thousands of applications running millions of transactions.

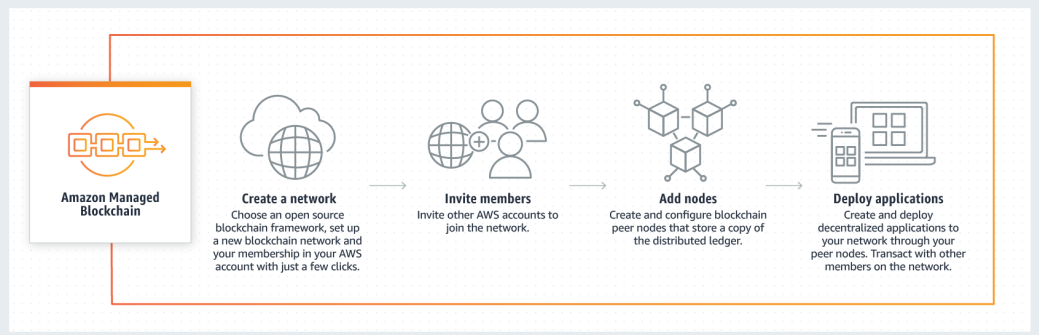
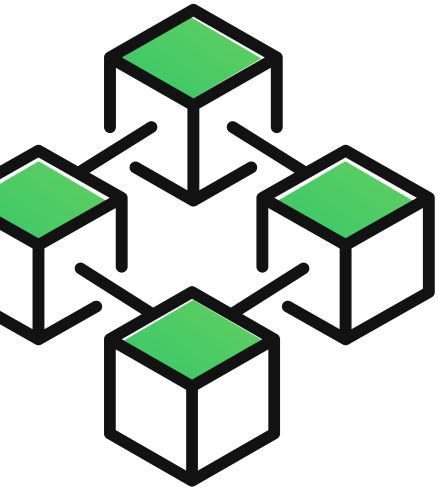


Fig 3 Amazon managed Blockchain



Managed Blockchain manages certificates and track operational metrics, such as usage of compute, memory, and storage resources. With Managed Blockchain's voting API, network participants can add or remove members. When a network member requires additional capacity for creating and validating transactions, the member can quickly add a new node using Managed Blockchain's APIs.

In addition, Managed Blockchain provides a selection of instance types that comprise varying combinations of CPU and memo. This gives you the flexibility to choose the appropriate mix of resources to support your nodes. Users pay according to their usage and do not worry about any upfront costs for infrastructure.

Managed Blockchain supports two popular blockchain frameworks, Hyperledger Fabric and Ethereum. Hyperledger Fabric is well-suited for applications that require stringent privacy and permission controls and with a known set of members. For example, this might include a financial application where certain trade-related data is only shared between a subset of the network (only the banks that are part of the trade).

Ethereum is well suited for highly distributed blockchain networks where transparency of data for all members is important. Each transaction is visible to all the members of the network. For example, this might include a customer loyalty blockchain application that allows any network retailer to verify user activity by broadcasting the transaction to all members.

Managed Blockchain is now generally available for Hyperledger Fabric.

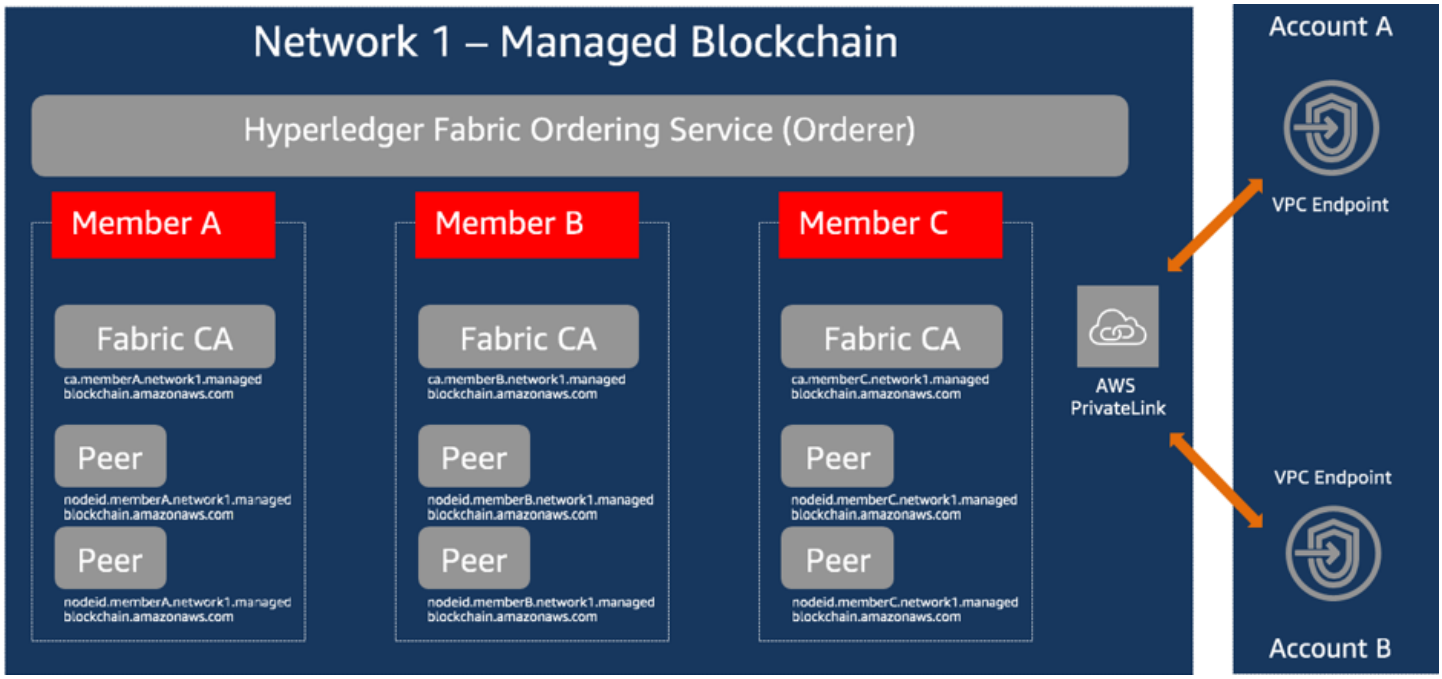


Fig 4 Hyperledger Fabric Ordering Service

AWS Managed Blockchain creates a network and manages its key components:

- The ordering service
- Members
- Hyperledger Fabric's certificate authority (CA) for each member in the network
- Peer nodes for the members

A blockchain network is a peer to peer network running a decentralized blockchain framework. Each network includes one or more members, which are unique identities in the network. Each member on the network can easily create their own peer nodes that come with a variety of compute and storage options.

Each member will have the Hyperledger Fabric CA. The Hyperledger Fabric CA provides a number of certificate services that relate to user enrolment, transactions invoked on the blockchain, and TLS-secured connections between users or components of the blockchain.

The peer nodes of each member interact to run smart contracts known as chain code in Hyperledger Fabric. They create and endorse transactions proposed in the network and store a local copy of the ledger.

Members define the rules in the endorsement process based on their application's business logic. For example, in a trade finance application, the bank for the exporting party wants to verify that the importing party has necessary funds before endorsing the transaction. To configure blockchain applications on peer nodes and to interact with other network resources, members use a client configured with the AWS CLI or SDK. Those network resources could include the certificate authority, ordering service, and peer nodes.

AWS Managed Blockchain provides endpoints to access these services, which can be accessed via an AWS PrivateLink endpoint. AWS Managed Blockchain ordering service is a component that ensures delivery of transactions across the blockchain network. AWS has built the ordering service using the underlying technology as Amazon QLDB. There is now an immutable change log that accurately maintains the complete history of all transactions in the blockchain network ensuring that you durably save this data.

A ledger with centralized trust

Some customers need a centralized ledger that records all changes or transactions and maintains an immutable record of these changes. This ledger is owned by a single trusted entity and is shared with any number of organizations that are working together. To do this today, customers can use relational databases like Oracle, MySQL, Postgres etc, or they can use the ledger technology in one of the open sources blockchain frameworks. Neither solution is optimal. Relational databases are not built for immutable, cryptographically verifiable ledger entries, so customers must build custom audit tables and audit trails. There is no way to verify that no unintended changes were made to the data. Using the ledger in a blockchain framework may give customers an immutable history of data changes, but comes at the cost of the heavy lifting to set up a full blockchain network with at least two nodes and all of the associated access control configuration. Because there are limited database Application Programming Interfaces (APIs) within the blockchain frameworks it is challenging to create tables, index, and query data. Finally, blockchain frameworks are decentralized and require consensus from members in the network before committing new transactions to the shared ledger, which significantly slows ledger performance.

Amazon QLDB makes it easy to understand how application data has changed over a period of time and reduces the need to build complicated audit functionality within the application. An Amazon QLDB journal is an immutable log where transactions are appended as blocks of data. After a transaction gets written as a block into the journal it cannot be changed or deleted if it becomes a permanent record.

These blocks are also hash-chained together using cryptography (SHA-256). This allows you to verify and show the proof of your data's integrity. This transaction then gets updated in the Current State table, which always keeps the current value of the data. Finally, the transaction gets indexed in the history table, which can be queried to track how the data has changed over time.

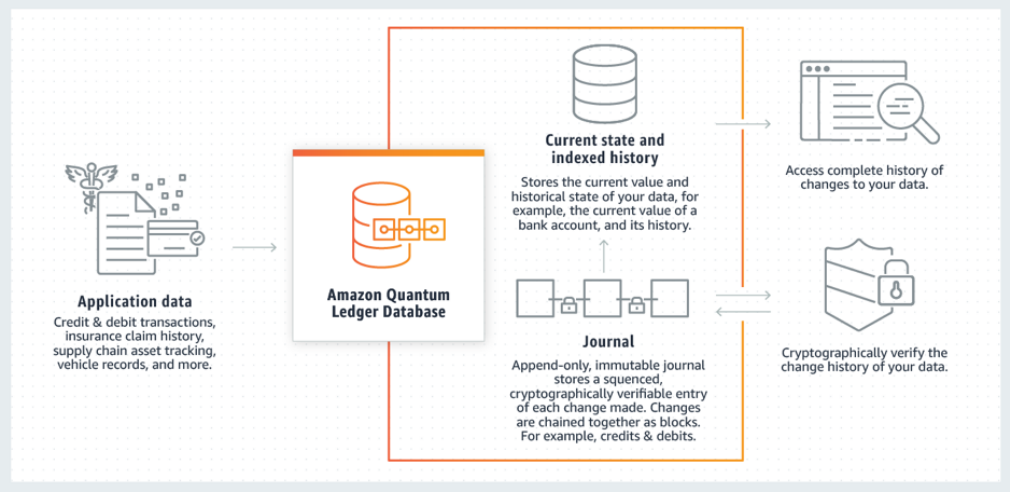


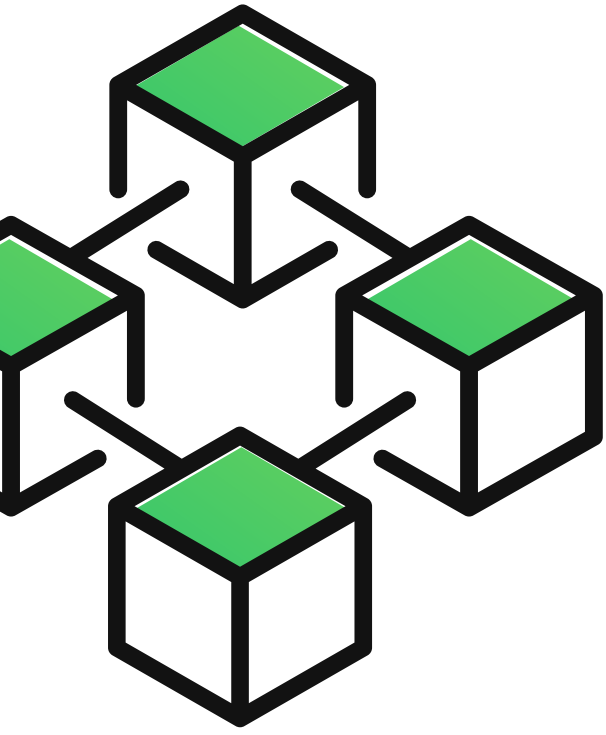
Fig 5. Amazon Quantum Ledger Database Source: AWS

brillio

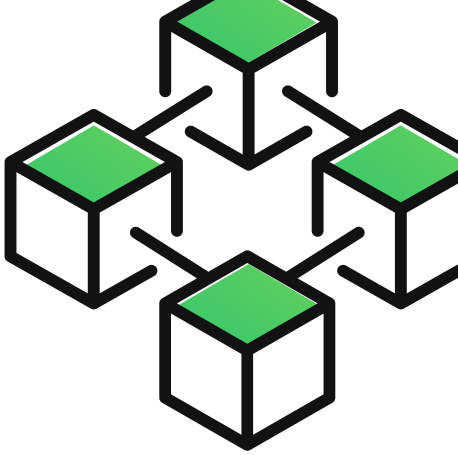
Making modern ledgers available to everyone

Blockchain and ledger technology has the potential to dramatically improve many types of transactions. However, if it is not accessible to everyone many great ideas may never be realized.

With AWS Managed Blockchain and Amazon QLDB anyone can create a ledger that is transparent, immutable, and cryptographically verifiable.



brillio



Summary

These two services AWS Managed Blockchain and Amazon QLDB help customers to solve their challenges on building the blockchain network. While the QLDB is designed to “provide transparent, immutable, and cryptographically verifiable log[s] of transactions that will be overseen by a central trusted authority” the Amazon Managed Blockchain will work with the QLDB to help its customers manage and adjust to a scalable blockchain network.

Author

Palanisamy Chellappan

With over 12 years of experience, as an Enterprise Architect, Palanisamy brings expertise in building the enterprise applications in Blockchain, AWS Cloud and Azure.