Lesson 2b: DLT Architecture

Module A: Blockchain Foundations

BSc Blockchain & Cryptocurrency

University Course

2025

Learning Objectives

By the end of this lesson, you will be able to:

1. Describe the structure and components of a blockchain block
2. Explain Merkle trees and their role in transaction verification
3. Differentiate between node types: full nodes, light nodes, miners, validators
4. Contrast permissioned and permissionless blockchain architectures

Prerequisites: L02a - DLT Fundamentals

This is Part 2 of 2 on DLT concepts. Part 1 (L02a) covered DLT definition and Byzantine consensus.

The Problem: How do we structure data for trustless verification?

Part 2/2: DLT Architecture (Integration)

The Challenge
How do we structure blocks and verify transactions efficiently across thousands of nodes without a central database?

Why It Matters

- Block structure determines security properties and performance limits
- Bitcoin's block size debate (2015-2017) led to SegWit upgrade and Lightning Network

What We Need

- Trustless verification mechanism
- Efficient data structures for tamper-evident storage at scale

The Cryptoeconomics Question
Coordinating without trusted intermediaries

Today's lesson: How DLT architecture addresses this challenge through Merkle trees and node design

Anatomy of a Block: Overview

A blockchain block contains two main parts:

1. **Block Header** (80 bytes in Bitcoin)
   - Metadata about the block
   - Links to previous block
   - Proof-of-Work evidence
2. **Block Body** (variable size)
   - Transaction data
   - Can contain hundreds to thousands of transactions

Block Size Limits:

- Bitcoin: 1 MB (base block) to 4 MB (with SegWit)
- Ethereum: Dynamic gas limit ($\approx$ 30M gas, $\approx$ 15 MB)
- Bitcoin Cash: 32 MB

Block Header Components

Bitcoin Block Header Fields (80 bytes):

1. **Version** (4 bytes): Block version number (for protocol upgrades)
2. **Previous Block Hash** (32 bytes): Hash of the previous block header
3. **Merkle Root** (32 bytes): Hash of all transactions in this block
4. **Timestamp** (4 bytes): Unix epoch time (seconds since Jan 1, 1970)
5. **Difficulty Target** (4 bytes): Required difficulty for PoW
6. **Nonce** (4 bytes): Counter used for PoW mining

Key Insight:

$$\text{Block Hash} = \text{SHA256}(\text{SHA256}(\text{Header}))$$

The hash must be below the difficulty target for the block to be valid.

Merkle Trees: Efficient Transaction Verification

Merkle Tree Definition

A **Merkle tree** is a binary hash tree where each leaf node is a hash of a transaction, and each non-leaf node is a hash of its two children.

Construction Process:

1. Hash each transaction: $H(Tx_1), H(Tx_2), ..., H(Tx_n)$
2. Pair hashes and hash again: $H(H(Tx_1)||H(Tx_2))$
3. Repeat until single root hash (Merkle Root)

Benefits:

- **Compact Proof**: Verify transaction inclusion with $O(\log n)$ hashes
- **Efficient Storage**: Light clients store only 80-byte headers
- **Integrity**: Any transaction change alters Merkle Root

Example: Prove transaction in block with 1,000 tx requires only 10 hashes ($\log_2 1000 \approx 10$)

Merkle Proof Example

Scenario: Block contains 8 transactions, prove Tx_3 is included

Tree Structure:

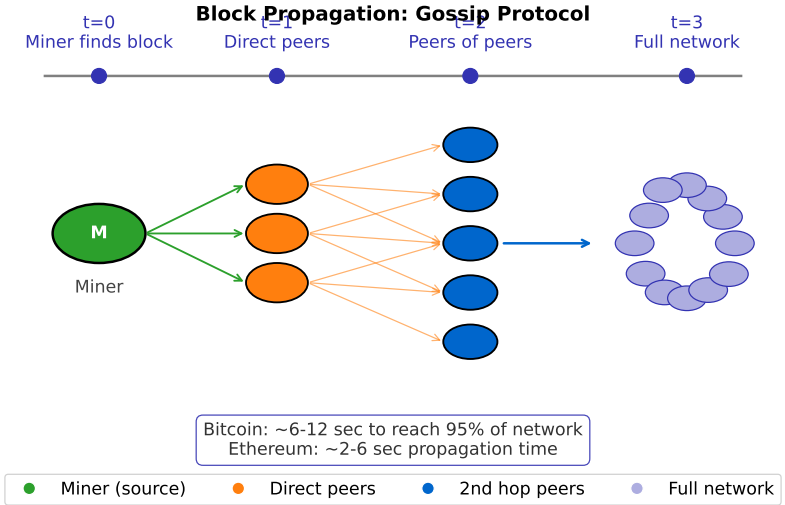
- Level 3 (Root): $R = H(AB||CD)$ where $AB = H(A||B)$, $CD = H(C||D)$
- Level 2: $A = H(Tx_1||Tx_2)$, $B = H(Tx_3||Tx_4)$, etc.
- Level 1: Tx_1, Tx_2, Tx_3, Tx_4, ..., Tx_8

Merkle Proof for Tx_3:

1. Verifier has: Block header with Merkle Root R
2. Prover provides: Tx_3, $H(Tx_4)$, $H(A)$, $H(CD)$
3. Verifier computes: $H(Tx_3)$, then $B = H(H(Tx_3)||H(Tx_4))$
4. Then: $AB = H(H(A)||B)$, finally $R' = H(AB||H(CD))$
5. If $R' = R$, then Tx_3 is proven to be in the block

Only 3 hashes needed instead of downloading all 8 transactions

Block Propagation via Gossip Protocol



Timeline: Miner broadcasts to peers, who propagate to their peers, until full network coverage.

Block Propagation Details

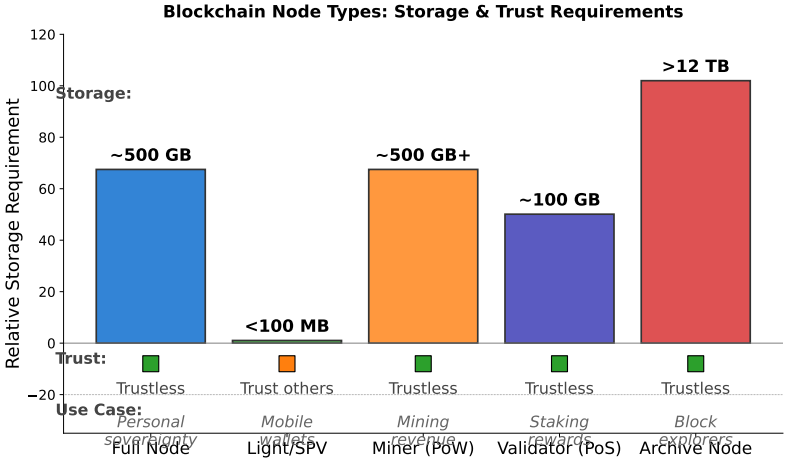
How New Blocks Spread Through Network:

1. Miner finds valid block (solves PoW puzzle)
2. Miner broadcasts block to connected peers ($\approx$ 8-125 peers)
3. Peers validate block:
   - Check block hash meets difficulty target
   - Verify all transactions are valid
   - Ensure no double-spends
4. Valid block propagated to peers' peers (gossip protocol)
5. Process repeats until entire network has the block

Propagation Time:

- Bitcoin: $\approx$ 6-12 seconds to reach 95% of network
- Ethereum (pre-merge): $\approx$ 2-6 seconds
- Faster propagation reduces orphan block rate

Blockchain Node Types: Storage and Trust



Trade-off: Higher storage requirements enable trustless verification; lower requirements require trusting others.

Full Nodes

Full Node Definition

A **full node** downloads and validates the entire blockchain history, enforcing all consensus rules without relying on third parties.

Responsibilities:

- Store complete blockchain (Bitcoin: $\approx$ 600 GB, Ethereum: $\approx$ 2 TB)
- Validate all blocks and transactions independently
- Relay valid transactions and blocks to peers
- Serve blockchain data to light clients

Benefits:

- Maximum security (trust no one)
- Support network decentralization
- Can verify historical transactions

Requirements: $\approx$ 500 GB storage, $\approx$ 5 GB/day bandwidth, moderate CPU

Light Nodes (SPV Nodes)

Simplified Payment Verification (SPV)

Described in Bitcoin whitepaper: Light nodes download only block headers ($\approx$ 80 bytes each) instead of full blocks, relying on Merkle proofs to verify transactions.

Operation:

- Download all block headers ($\approx$ 50 MB for Bitcoin)
- Request Merkle proofs from full nodes for relevant transactions
- Verify transaction inclusion in block
- Assume longest chain is valid

Trade-offs:

+ Low storage ($<$ 100 MB vs. 500 GB)
+ Suitable for mobile devices
- Must trust full nodes for transaction data
- Vulnerable to bloom filter privacy leaks

Miner Nodes (PoW)

Specialized Nodes in Proof-of-Work Blockchains

Functions:

1. Collect unconfirmed transactions from mempool
2. Construct candidate block (select transactions based on fees)
3. Attempt to find valid nonce (compute millions of hashes/second)
4. Broadcast valid block if found
5. Receive block reward + transaction fees

Requirements:

- Specialized hardware (ASICs for Bitcoin, GPUs for some altcoins)
- High electricity consumption (Bitcoin: $\approx$ 150 TWh/year globally)
- Cooling infrastructure
- High-bandwidth internet connection

Economics:

- Bitcoin block reward: 3.125 BTC (post-April 2024 halving)
- Break-even depends on: Hash rate, electricity cost, BTC price

Validator Nodes (PoS)

Specialized Nodes in Proof-of-Stake Blockchains

Functions:

1. Stake cryptocurrency as collateral (Ethereum: 32 ETH minimum)
2. Selected to propose/attest blocks based on stake
3. Earn rewards for honest participation
4. Risk slashing (stake confiscation) for malicious behavior

Ethereum Validator Requirements:

- 32 ETH staked (locked)
- Run validator client 24/7 ($>$ 99% uptime)
- Consumer-grade hardware (4 GB RAM, 100 GB SSD)
- Stable internet (10 Mbps)

Comparison to Mining:

+ 99.95% less energy consumption
+ Lower hardware requirements
- Capital locked (opportunity cost)
- Slashing risk if offline or malicious

Archive Nodes

Full Historical State Storage

- Store every state of the blockchain at every block height
- Enable queries like: "What was Alice's balance at block 1,000,000?"
- Required for block explorers (Etherscan, Blockchain.com)
- Needed for advanced analytics and forensics

Storage Requirements:

- Bitcoin archive node: $\approx$ 500 GB (same as full node, UTXO model)
- Ethereum archive node: $\approx$ 12 TB (account model stores all states)
- Grows continuously (Ethereum: $+1$ TB/year)

Use Cases:

- Block explorers
- Tax reporting services
- Academic research
- Forensic analysis

Node Type Comparison

Node Type	Storage	Validation	Trust Model	Use Case
Full Node	$\approx$ 500 GB	All blocks & txs	Trustless	Personal sovereignty
Light/SPV	$<$ 100 MB	Headers only	Trust full nodes	Mobile wallets
Miner (PoW)	$\approx$ 500 GB +	Full validation	Trustless	Mining revenue
Validator (PoS)	$\approx$ 100 GB	Full validation	Trustless	Staking rewards
Archive Node	$>$ 10 TB	Full + history	Trustless	Analytics, explorers

Decentralization Trade-off:

- More full nodes = more decentralization
- High storage requirements = fewer full nodes
- Blockchain scalability trilemma: Security, Decentralization, Scalability

Permissionless Blockchains (Public)

Definition

Permissionless (public) blockchains allow anyone to join, read, write, and participate in consensus without approval from a central authority.

Characteristics:

- Open participation (anyone can run a node)
- Pseudonymous identities (addresses, not real names)
- Transparent transaction history (all data public)
- Censorship resistant
- Token-based incentives (mining/staking rewards)

Examples: Bitcoin, Ethereum, Cardano, Solana

Best For: Global, trustless applications (DeFi, NFTs), censorship-resistant systems

Permissioned Blockchains (Private/Consortium)

Definition

Permissioned blockchains restrict participation to verified entities. Access to read, write, or validate is controlled by administrators.

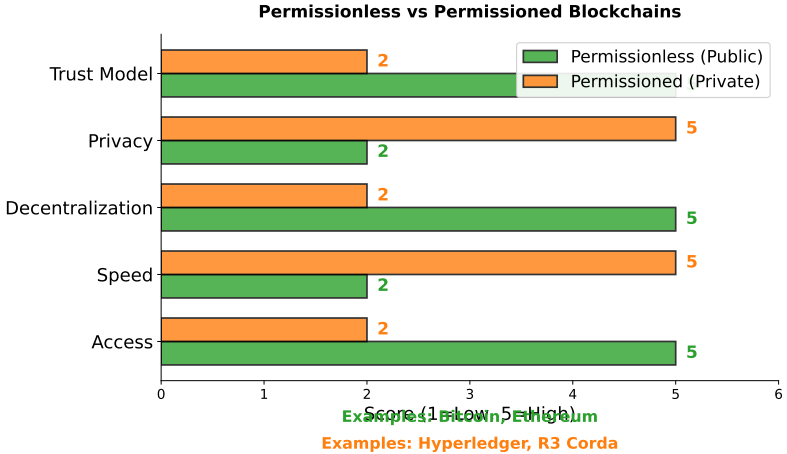
Characteristics:

- Controlled participation (invitation-only)
- Known identities (KYC/AML compliance)
- Selective transparency (private channels possible)
- Efficient consensus (no mining, BFT-based)
- No native cryptocurrency (optional)

Examples: Hyperledger Fabric, R3 Corda, JP Morgan Quorum

Best For: Enterprise consortia (supply chain, trade finance), regulatory compliance

Permissioned vs Permissionless: Key Dimensions



Trade-off: Permissionless maximizes decentralization/trust; Permissioned maximizes speed/privacy.

Comparison: Permissionless vs. Permissioned

Permissionless (Public)

Advantages:

- Maximum decentralization
- Censorship resistant
- Network effects (large user base)
- Innovation without permission

Disadvantages:

- Slower throughput (5-30 TPS)
- Higher costs (gas fees)
- Energy intensive (PoW)
- No privacy by default

Permissioned (Private)

Advantages:

- High throughput (1,000+ TPS)
- Low/no transaction fees
- Energy efficient (BFT)
- Configurable privacy

Disadvantages:

- Central point of control
- Limited network effects
- Potential for censorship
- Requires trust in admins

Hybrid models exist: Public with private sidechains (e.g., Polygon for Ethereum)

When to Choose Each Type

Criterion	Permissionless	Permissioned
Trust Model	No trust required	Parties partially trust each other
Participants	Unknown, global	Known, verified entities
Governance	Decentralized (hard forks)	Centralized/consortium voting
Performance	5-30 TPS	1,000-10,000+ TPS
Privacy	Pseudonymous, transparent	Configurable, private channels
Use Cases	DeFi, NFTs, payments	Supply chain, trade finance, CBDCs

Problem Solved? DLT Architecture

The Original Problem

How do we structure data for trustless verification across thousands of nodes?

How DLT Architecture Solves It

- Merkle trees enable O(log n) verification instead of downloading all transactions
- Block headers separate metadata from transaction data (80 bytes vs. MB)
- Full nodes independently validate entire history without trusting anyone

Remaining Limitations

- Block size limits transaction throughput (Bitcoin: 7 TPS, Ethereum: 30 TPS)
- Storage requirements create centralization pressure (600 GB+ for full nodes)

Open Questions

- Optimal balance between full nodes (decentralization) and light clients (accessibility)?
- Risk: As storage grows, fewer users can verify independently

DLT architecture enables trustless verification but introduces scalability-decentralization trade-offs

Key Takeaways

What You Should Remember:

1. **Block Structure**: Header (80 bytes) + Body (transactions), linked via cryptographic hashes
2. **Merkle Trees**: Enable efficient transaction verification ($O(\log n)$ proof size)
3. **Node Types**:
   - Full (trustless), Light (trust others)
   - Miner/Validator (consensus), Archive (historical queries)
4. **Permissioned vs. Permissionless**: Public for trustless apps, private for enterprise consortia

Next: L03 - Cryptographic Hash Functions

Discussion Questions

Consider and discuss:

1. **Decentralization Trade-offs**: At what point does storage cost compromise decentralization?
2. **Node Selection**: When would you run a full node vs. use a light client?
3. **Enterprise Blockchain**: Why do enterprises often prefer permissioned chains?

Thank you

Questions?

Continue to L03: Cryptographic Hash Functions

Quiz Questions (1/2)

Q1. What is the size of a Bitcoin block header?
A) 32 bytes B) 80 bytes C) 256 bytes D) 1 MB
Answer: B – Bitcoin block headers are exactly 80 bytes (6 fields).

Q2. What is the purpose of the Merkle Root in a block header?
A) Identify previous block B) Prove all transactions included C) Set mining difficulty D) Store timestamp
Answer: B – Merkle Root commits to all transactions, enabling efficient verification.

Q3. How many hashes are needed to verify a transaction in a block with 1,024 transactions using a Merkle proof?
A) 5 B) 10 C) 512 D) 1,024
Answer: B – $\log_2 1024 = 10$ hashes needed for Merkle proof.

Q4. What approximate storage is required for a Bitcoin full node?
A) 50 MB B) 10 GB C) 600 GB D) 12 TB
Answer: C – Bitcoin full nodes require approximately 600 GB (as of 2025).

Q5. What does SPV stand for in the context of light nodes?
A) Secure Payment Validation B) Simplified Payment Verification C) Shared Proof Verification D) Synchronized Peer Validation
Answer: B – SPV (Simplified Payment Verification) described in Bitcoin whitepaper.

Quiz Questions (2/2)

Q6. What is the minimum stake required to run an Ethereum validator?
A) 1 ETH B) 10 ETH C) 32 ETH D) 100 ETH
Answer: C – Ethereum requires exactly 32 ETH to activate a validator.

Q7. Which node type stores every historical state at every block height?
A) Full node B) Light node C) Archive node D) Miner node
Answer: C – Archive nodes store complete historical state (e.g., 12 TB for Ethereum).

Q8. Which blockchain type typically achieves higher throughput (TPS)?
A) Permissionless public chains B) Permissioned private chains C) Both equal D) Depends only on hardware
Answer: B – Permissioned chains achieve 1,000+ TPS vs. 5-30 TPS for public chains.

Q9. What protocol do Bitcoin nodes use to propagate new blocks?
A) HTTP B) FTP C) Gossip protocol D) TCP handshake
Answer: C – Gossip protocol: nodes relay to peers, who relay to their peers.

Q10. Which field in the Bitcoin block header links blocks together?
A) Merkle Root B) Nonce C) Previous Block Hash D) Timestamp
Answer: C – Previous Block Hash (32 bytes) creates the cryptographic chain.