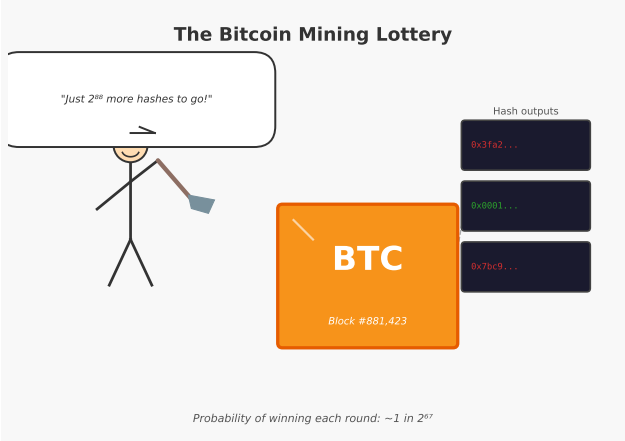
L03: Bitcoin – Technical Deep Dive

BSc Blockchain Course

Digital Finance

What If Money Could Run on Math Instead of Trust?



Imagine sending money to anyone on Earth – no bank, no wire transfer, no permission needed. In 2009, Satoshi Nakamoto made this real with Bitcoin. Today we open the hood: UTXOs, blocks, mining, scripts, and the fee market.

This cartoon frames the central question of Lesson 3: how does trustless digital money actually work?

Learning Objectives

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

1. **Explain** the UTXO model and how it differs from account-based systems (Ethereum). *[Understand]*
2. **Describe** Bitcoin block structure, header fields, and the mining process. *[Understand]*
3. **Calculate** difficulty adjustments and predict mining reward given a halving epoch. *[Apply]*
4. **Analyze** the fee market, mempool dynamics, and transaction priority mechanisms. *[Analyze]*
5. **Explain** chain selection rules, fork resolution, and confirmation security trade-offs. *[Evaluate]*

Bloom's levels covered: Understand, Apply, Analyze, Evaluate

Prerequisites: L02 Cryptographic Foundations (hashing, digital signatures, Merkle trees).

The First Cryptocurrency

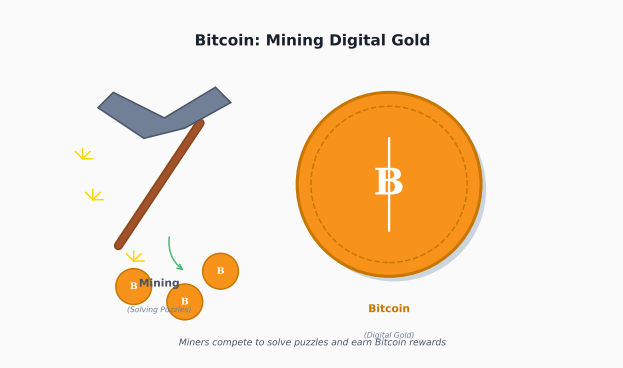
Bitcoin introduced the world to decentralized digital money. Understanding its design reveals the core principles that all later cryptocurrencies build upon.

Core components:

- **Transactions:** Transfer value between addresses
- **Blocks:** Bundle transactions with proof-of-work
- **Chain:** Linked blocks forming immutable history
- **Network:** P2P nodes validating and relaying
- **Miners:** Create new blocks, earn rewards

Design philosophy:

- Simple, conservative, secure
- Prioritize decentralization over features
- Slow, deliberate upgrades



- **What you see:** Bitcoin's core architecture – transactions, blocks, and the chain that links them.
- **Key pattern:** Every component reinforces decentralization: no single point of failure.
- **Takeaway:** "Be your own bank" – Bitcoin enables self-custody of digital value

What Happens When You Lose Your Keys?

Take a moment and consider what "be your own bank" really means – including the terrifying downside.

Three cautionary tales:

- **Satoshi Nakamoto** mined roughly **1 million BTC** in Bitcoin's early days. Those coins have *never moved*. If Satoshi lost the keys, that is over $60 billion – permanently inaccessible.
- **James Howells** (Wales, 2013) accidentally threw away a hard drive containing 8,000 BTC. The drive sits in a landfill. Howells has spent years negotiating with the local council to dig it up. Current value: over $500 million.
- **Stefan Thomas** (San Francisco) stored 7,002 BTC on an encrypted IronKey USB drive. He has **2 password attempts left** out of 10 before the drive wipes itself permanently.

The core trade-off: In traditional banking, you can reset your password. In Bitcoin, there is **no "forgot password" button**. If you lose your private key, your coins are gone forever – no customer support, no court order, no recovery.

Question for the class: Is this a bug or a feature?

An estimated 3–4 million BTC (15–20% of supply) are permanently lost due to forgotten keys and discarded hardware.

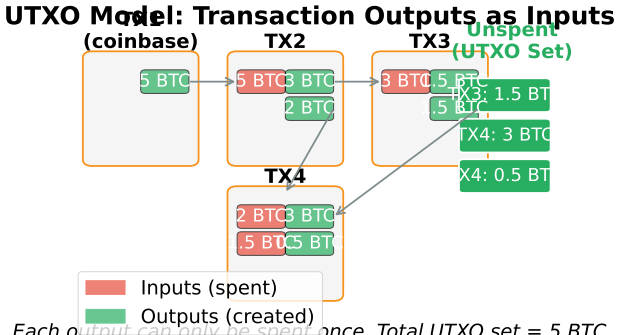
UTXO Model Explained

UTXO = Unspent Transaction Output. This is the fundamental building block of Bitcoin's value-tracking system.

How it works:

- Transactions **consume** existing UTXOs (inputs) and **create** new ones (outputs)
- Each UTXO can be spent **exactly once** – like a banknote
- Your "balance" = the sum of all UTXOs your private key can unlock
- There is no "account" object anywhere in Bitcoin

Analogy: Think of UTXOs as physical coins and bills. If you have a $20 bill and want to pay $15, you hand over the $20 and get $5 back as "change." Bitcoin works identically – the entire UTXO is consumed, and a new UTXO is created for the change.



Each output can only be spent once. Total UTXO set = 5 BTC

- **What you see:** UTXOs being consumed as inputs and new UTXOs created as outputs.
- **Key pattern:** Inputs are always fully spent – any leftover value must be sent back as change.
- **Takeaway:** Bitcoin tracks discrete units of value (UTXOs), not running balances.

The UTXO set (all currently unspent outputs) is roughly 80–90 million entries as of 2025.

UTXO Flow: Inputs, Outputs, and Change

A single Bitcoin transaction can combine **multiple inputs** and produce **multiple outputs**. This allows flexible payment splitting.

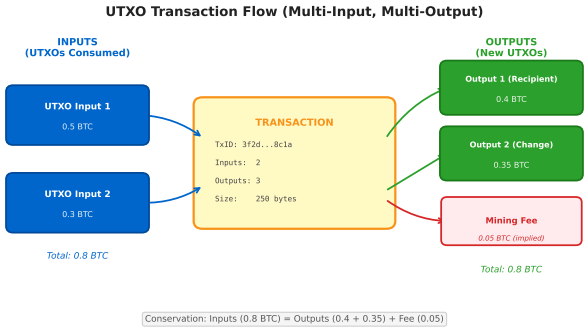
Example transaction:

- **Input 1:** 0.5 BTC (from a previous tx)
- **Input 2:** 0.3 BTC (from another previous tx)
- **Output 1:** 0.7 BTC (payment to recipient)
- **Output 2:** 0.0999 BTC (change back to sender)
- **Implicit fee:** 0.0001 BTC (claimed by miner)

Key rule:

$$\sum \text{inputs} = \sum \text{outputs} + \text{fee}$$

If you forget the change output, the entire difference becomes the miner's fee – an expensive mistake.



- **What you see:** Multiple UTXOs flowing into a transaction and new UTXOs emerging.
- **Key pattern:** Every satoshi is accounted for – inputs fully consumed, outputs explicitly created.
- **Takeaway:** Wallets manage UTXO selection automatically, but understanding the flow prevents costly errors.

UTXO vs Account Model: Two Philosophies

Bitcoin and Ethereum take fundamentally different approaches to tracking who owns what. Neither is objectively better – they optimize for different goals.

Dimension	UTXO Model (Bitcoin)	Account Model (Ethereum)
What is tracked	Individual outputs (discrete coins)	Balance per address
Privacy	Better – new address per transaction	Worse – reuses same address
Parallel validation	Yes – independent UTXO checks	No – sequential nonce required
Wallet complexity	Higher – must manage UTXO selection	Lower – simple balance math
Smart contracts	Limited (Bitcoin Script)	Full (Solidity, EVM)
State size	UTXO set grows with usage	Account trie grows with usage
Double-spend check	Check if UTXO exists and is unspent	Check nonce ordering

Key insight: Bitcoin's UTXO model is optimized for **value transfer and privacy**. Ethereum's account model is optimized for **programmable state and smart contracts**. Each design choice cascades through the entire system.

Cardano and Ergo use an "extended UTXO" model that adds smart contract support to the UTXO paradigm.

Bitcoin Block Structure

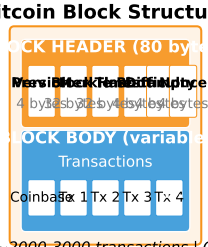
Every Bitcoin block has two parts: a compact **header** (always 80 bytes) and a variable-size **body** containing transactions.

Header fields (80 bytes total):

- **Version** (4 B): Protocol version
- **Prev Block Hash** (32 B): Links to parent
- **Merkle Root** (32 B): Fingerprint of all txs
- **Timestamp** (4 B): Block creation time
- **Bits** (4 B): Difficulty target
- **Nonce** (4 B): Mining solution counter

Block body:

- Coinbase tx (creates new BTC + collects fees)
- Regular transactions (1,000–3,000 typical)
- Size: 1.5–2 MB with SegWit



Average block: ~1-2 MB | ~2000-3000 transactions | Created every ~10 minutes

- **What you see:** The anatomy of a Bitcoin block – header fields linked to transaction body via Merkle root.
- **Key pattern:** The header is tiny (80 bytes) but commits to the entire block content through the Merkle root hash.
- **Takeaway:** Only the 80-byte header is hashed for proof-of-work – miners never hash the full block.

Block Hash = SHA256(SHA256(header)). Changing any transaction changes the Merkle root, which changes the header hash.

Transaction Anatomy

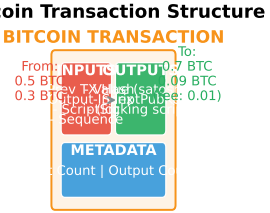
A Bitcoin transaction is a signed data structure that moves value from one set of UTXOs to another. Every transaction is publicly visible on the blockchain.

Structure:

- **Version:** Transaction format version
- **Inputs:** Which UTXOs are being spent
- **Outputs:** Where the value goes
- **Locktime:** Earliest time/block for inclusion
- **Witness** (SegWit): Signature data

Typical sizes:

- 1-in, 2-out (P2WPKH): ∼140 vbytes
- 2-in, 2-out (P2WPKH): ∼208 vbytes
- Legacy P2PKH: ∼226 bytes



Inputs unlock funds | Outputs create new UTXOs | Fee = Sum(inputs) - Sum(outputs)

- **What you see:** Inputs referencing previous outputs and new outputs being created with locking scripts.
- **Key pattern:** Inputs unlock (ScriptSig) and outputs lock (ScriptPubKey) – a chain of cryptographic custody.
- **Takeaway:** Every satoshi has a traceable history back to the coinbase transaction that created it.

Inputs reference previous outputs; outputs create new UTXOs. The difference is the miner's fee.

Transaction Components: A Closer Look

Inputs (spending existing UTXOs):

- **Previous tx hash:** Which transaction created the UTXO
- **Output index:** Which output of that transaction (0, 1, 2...)
- **ScriptSig:** The unlocking script (signature + public key)
- **Sequence number:** Used for RBF and timelocks

Outputs (creating new UTXOs):

- **Value:** Amount in satoshis (1 BTC = 100,000,000 satoshis)
- **ScriptPubKey:** The locking script – defines who can spend this output

Transaction Components: A Closer Look

Inputs (spending existing UTXOs):

- **Previous tx hash:** Which transaction created the UTXO
- **Output index:** Which output of that transaction (0, 1, 2...)
- **ScriptSig:** The unlocking script (signature + public key)
- **Sequence number:** Used for RBF and timelocks

Outputs (creating new UTXOs):

- **Value:** Amount in satoshis (1 BTC = 100,000,000 satoshis)
- **ScriptPubKey:** The locking script – defines who can spend this output

Transaction Fee:

$$\text{Fee} = \sum \text{Input Values} - \sum \text{Output Values}$$

The fee is *implicit* – there is no "fee" field. Whatever value is left over after outputs are created goes to the miner.

Transaction Components: A Closer Look

Inputs (spending existing UTXOs):

- **Previous tx hash:** Which transaction created the UTXO
- **Output index:** Which output of that transaction (0, 1, 2...)
- **ScriptSig:** The unlocking script (signature + public key)
- **Sequence number:** Used for RBF and timelocks

Outputs (creating new UTXOs):

- **Value:** Amount in satoshis (1 BTC = 100,000,000 satoshis)
- **ScriptPubKey:** The locking script – defines who can spend this output

Transaction Fee:

$$\text{Fee} = \sum \text{Input Values} - \sum \text{Output Values}$$

The fee is *implicit* – there is no "fee" field. Whatever value is left over after outputs are created goes to the miner.

Worked example: Input: 0.05 BTC. Output 1: 0.03 BTC (payment). Output 2: 0.0199 BTC (change). Fee = $0.05 - 0.03 - 0.0199 = 0.0001$ BTC = 10,000 satoshis.

---

1 BTC = 100,000,000 satoshis. The smallest unit is named after Bitcoin's creator.

Bitcoin Script: A Stack-Based Language

Bitcoin uses a simple, **stack-based scripting language** to define spending conditions. It is intentionally **not Turing-complete**.

How a P2PKH script executes:

1. Push signature onto stack
2. Push public key onto stack
3. `OP_DUP`: duplicate the public key
4. `OP_HASH160`: hash the duplicate
5. `OP_EQUALVERIFY`: compare with address hash
6. `OP_CHECKSIG`: verify signature

If the stack ends with TRUE: the UTXO is unlocked and can be spent. Otherwise, the transaction is invalid.

Security benefit: No loops means no infinite execution, no gas fees needed, and deterministic verification.

P2PKH Script Execution (Pay-to-Public-Key-Hash)

ScriptPubKey (Locking): OP_DUP OP_HASH160 <PubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
ScriptSig (Unlocking): <Signature> <PublicKey>

- **What you see:** Step-by-step script execution showing the stack state at each opcode.
- **Key pattern:** The script is split: ScriptPubKey locks, ScriptSig unlocks. Both must execute together successfully.
- **Takeaway:** Bitcoin Script has about 100 opcodes, many disabled for safety – simplicity is a security feature.

Stack-based scripting: intentionally NOT Turing-complete. This prevents infinite loops and simplifies security analysis.

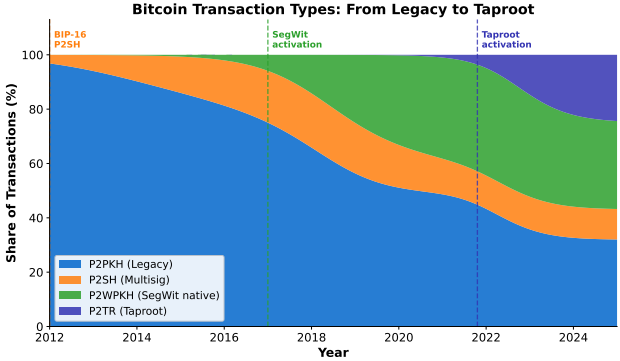
Script Types: From P2PKH to Taproot

Bitcoin's script types have evolved over 15 years, each generation adding efficiency and privacy.

Four generations:

1. **P2PKH** (2009): Pay to Public Key Hash
   Address starts with `1...`
   Original format, largest on-chain footprint
2. **P2SH** (2012, BIP 16): Pay to Script Hash
   Address starts with `3...`
   Enables multisig and complex conditions
3. **P2WPKH** (2017, SegWit): Native SegWit
   Address starts with `bc1q...`
   Smaller transactions, lower fees
4. **P2TR** (2021, Taproot): Pay to Taproot
   Address starts with `bc1p...`
   Schnorr signatures, best privacy

Why not Turing-complete? Prevents infinite loops, enables deterministic execution, and reduces the attack surface.



- **What you see:** The evolution of Bitcoin transaction types showing adoption share over time.
- **Key pattern:** Each new type gains adoption gradually – P2WPKH now dominates new transactions.
- **Takeaway:** Backward compatibility is preserved: old address types still work, but newer types save fees.

Mining Process: The SHA256 Lottery

Bitcoin mining is a **computational lottery**. Miners repeatedly hash block headers, searching for a result below the difficulty target.

Proof-of-Work algorithm:

1. **Collect** highest-fee transactions from the mempool
2. **Build** a candidate block with a coinbase transaction (mining reward + fees)
3. **Hash** the 80-byte header: $H = \text{SHA256}(\text{SHA256}(\text{header}))$
4. **Check:** Is $H <$ target? If yes, you win. If no, continue.
5. **Increment** the nonce (and extraNonce in coinbase) and repeat
6. **Broadcast** the winning block to the network

Current statistics (2025):

- Hash rate: $\sim$1 ZH/s (zettahash/s $= 10^{21}$ hashes/s)
- Energy consumption: $\sim$170 TWh/year
- Average block time: 10 minutes
- Nonce space (2^{32}) exhausted in under 1 second by modern ASICs

---

Mining is a lottery – more hash power = more tickets, but each individual attempt has the same tiny probability.

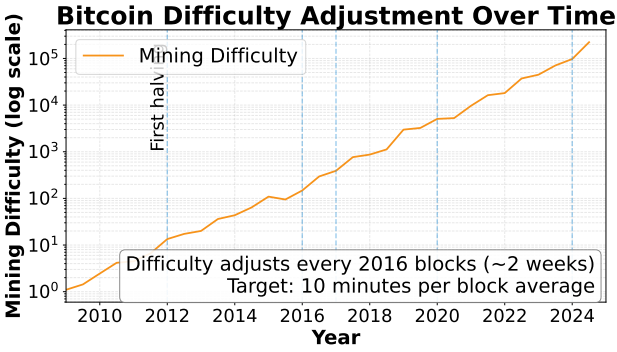
Difficulty Adjustment

Bitcoin's **difficulty adjustment** is an elegant self-regulating mechanism that maintains the 10-minute block target regardless of how much mining power joins or leaves the network.

How it works:

- Every **2,016 blocks** (∼2 weeks), the network measures how long those blocks actually took
- If blocks were too fast: difficulty **increases**
- If blocks were too slow: difficulty **decreases**
- Change is capped at **4x** per adjustment (up or down)

Why it matters: Without difficulty adjustment, a doubling of hash rate would halve the block time to 5 minutes, accelerating Bitcoin's emission schedule and undermining its monetary policy.



- **What you see:** Historical difficulty adjustments tracking hash rate growth over time.
- **Key pattern:** Difficulty has only ever gone significantly down during mining bans (e.g., China 2021).
- **Takeaway:** The adjustment mechanism has kept average block time remarkably close to 10 minutes for 16 years.

Adjusts every 2,016 blocks (∼2 weeks) to maintain the 10-minute target. A beautifully simple feedback loop.

Difficulty Algorithm: A Worked Example

Adjustment formula:

$$\text{New Difficulty} = \text{Old Difficulty} \times \frac{\text{Target Time}}{\text{Actual Time}}$$

where Target Time $= 2{,}016 \times 10$ min $= 20{,}160$ minutes (14 days).

Safety cap: Maximum 4x increase or 0.25x decrease per period.

Difficulty Algorithm: A Worked Example

Adjustment formula:

$$\text{New Difficulty} = \text{Old Difficulty} \times \frac{\text{Target Time}}{\text{Actual Time}}$$

where Target Time $= 2{,}016 \times 10$ min $= 20{,}160$ minutes (14 days).

Safety cap: Maximum 4x increase or 0.25x decrease per period.

Example 1 – blocks too fast:

- Last 2,016 blocks took 10,080 min (7 days instead of 14)
- Ratio: $\frac{20{,}160}{10{,}080} = 2.0$
- New Difficulty = Old Difficulty $\times$ 2.0 (difficulty **doubles**)
- Result: miners now need twice as many hashes on average to find a block

Difficulty Algorithm: A Worked Example

Adjustment formula:

$$\text{New Difficulty} = \text{Old Difficulty} \times \frac{\text{Target Time}}{\text{Actual Time}}$$

where Target Time $= 2{,}016 \times 10$ min $= 20{,}160$ minutes (14 days).

Safety cap: Maximum 4x increase or 0.25x decrease per period.

Example 1 – blocks too fast:

- Last 2,016 blocks took 10,080 min (7 days instead of 14)
- Ratio: $\frac{20{,}160}{10{,}080} = 2.0$
- New Difficulty = Old Difficulty $\times$ 2.0 (difficulty **doubles**)
- Result: miners now need twice as many hashes on average to find a block

Example 2 – blocks too slow:

- Last 2,016 blocks took 40,320 min (28 days instead of 14)
- Ratio: $\frac{20{,}160}{40{,}320} = 0.5$
- New Difficulty = Old Difficulty $\times$ 0.5 (difficulty **halves**)
- Result: blocks become easier to find, bringing the average back to 10 minutes

---

Self-adjusting system: maintains stability as hash rate changes. Unique to Bitcoin – most systems need manual tuning.

Mining Reward Schedule

Bitcoin's block reward **halves every 210,000 blocks** ($\sim$4 years), creating a predictable and declining emission schedule.

Halvings completed:

1. Nov 2012: 50 $\rightarrow$ 25 BTC
2. Jul 2016: 25 $\rightarrow$ 12.5 BTC
3. May 2020: 12.5 $\rightarrow$ 6.25 BTC
4. Apr 2024: 6.25 $\rightarrow$ 3.125 BTC

Worked example – reward at block 900,000:

Mining Reward Schedule

Bitcoin's block reward **halves every 210,000 blocks** ($\sim$4 years), creating a predictable and declining emission schedule.

Halvings completed:

1. Nov 2012: 50 $\rightarrow$ 25 BTC
2. Jul 2016: 25 $\rightarrow$ 12.5 BTC
3. May 2020: 12.5 $\rightarrow$ 6.25 BTC
4. Apr 2024: 6.25 $\rightarrow$ 3.125 BTC

Worked example – reward at block 900,000:

- Halving epoch $= \lfloor 900{,}000/210{,}000 \rfloor = 4$
- Reward $= 50/2^4 = 50/16 = 3.125$ BTC

Miner revenue:

$$\text{Revenue} = \text{Block Reward} + \sum \text{Tx Fees}$$

As block rewards shrink, fees must sustain the security budget.

21M Cap

- **What you see:** Block reward declining in a step function, with cumulative supply approaching the 21M cap.
- **Key pattern:** Each halving cuts the daily new supply in half – a built-in deflationary mechanism.
- **Takeaway:** Over 93% of all Bitcoin that will ever exist has already been mined.

Block reward halves every 210,000 blocks ($\sim$4 years). The last satoshi will be mined around year 2140.

Bitcoin Supply vs Fiat Money Printing

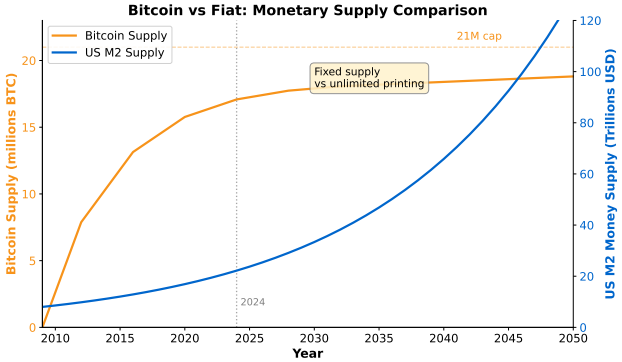
Bitcoin's halving table:

Period	Reward	Supply%
2009–2012	50 BTC	50%
2012–2016	25 BTC	75%
2016–2020	12.5 BTC	87.5%
2020–2024	6.25 BTC	93.75%
2024–2028	3.125 BTC	96.875%

Key facts:

- Total supply: 21,000,000 BTC (hard cap)
- Current supply: ∼20M BTC (95%)
- Last bitcoin mined: ∼year 2140

Contrast: The US M2 money supply grew from $4.6T (2000) to $21T (2024) – a 4.5x increase. Bitcoin's supply schedule is fixed in code.



- **What you see:** Bitcoin's supply curve (logarithmic, capped) compared to fiat money supply (exponential, uncapped).
- **Key pattern:** Bitcoin supply is convex and bounded; fiat supply is concave and unbounded.
- **Takeaway:** Predictable monetary policy with no central bank discretion – a feature, not a bug, to Bitcoin advocates.

Mining Profitability: Cost vs Reward

Mining is only profitable when revenue exceeds costs. The economics are brutally competitive.

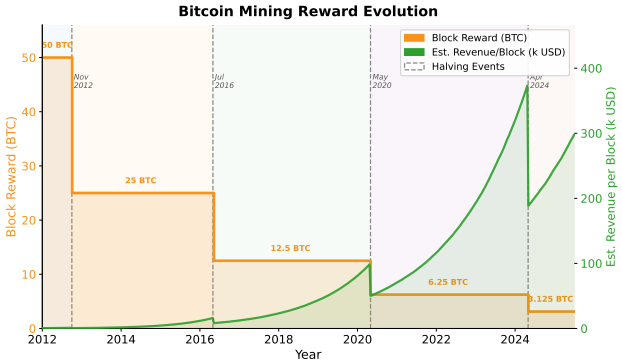
Revenue components:

- Block reward (currently 3.125 BTC)
- Transaction fees (∼0.1–0.5 BTC/block typical)

Cost components:

- **Electricity:** 60–70% of operating cost
- **Hardware:** ASIC miners ($2K–$15K each)
- **Cooling:** Significant in warm climates
- **Facility:** Rent, internet, maintenance

Breakeven electricity cost (2025): approximately $0.05–0.08/kWh depending on hardware generation. Miners with costs above $0.10/kWh are unprofitable at current BTC prices.



- **What you see:** Mining revenue vs cost curves across different electricity rates and BTC prices.
- **Key pattern:** Each halving compresses margins – only the most efficient miners survive.
- **Takeaway:** Mining is an arms race: hardware improves, difficulty rises, and margins converge toward zero.

Energy Consumption and the Mining Mix

Bitcoin mining: ∼**170 TWh/year** – comparable to Poland.

Energy mix (2024):

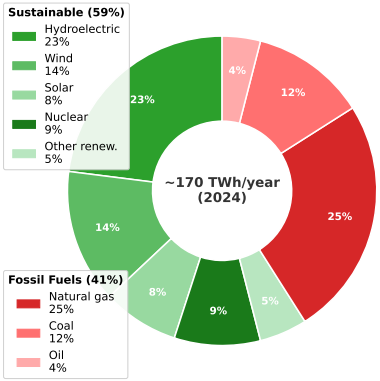
- **Sustainable:** ∼59% (hydro, wind, solar, nuclear)
- **Fossil fuels:** ∼41% (natural gas dominant)

Why miners seek cheap energy:

- Electricity = 60–70% of cost
- Cheapest sources: stranded renewables
- Co-locate with hydro dams, flared gas

The debate: "Wasted" energy or the cost of censorship-resistant money?

Bitcoin Mining Energy Mix (2024 Estimates)



- **What you see:** Energy source breakdown for mining.
- **Key pattern:** Sustainable share: 35% (2020) to 59% (2024).
- **Takeaway:** The energy mix matters more than total consumption.

Cambridge Centre for Alternative Finance provides the most cited estimates of Bitcoin electricity consumption.

Hash Rate History: From CPUs to Zettahash

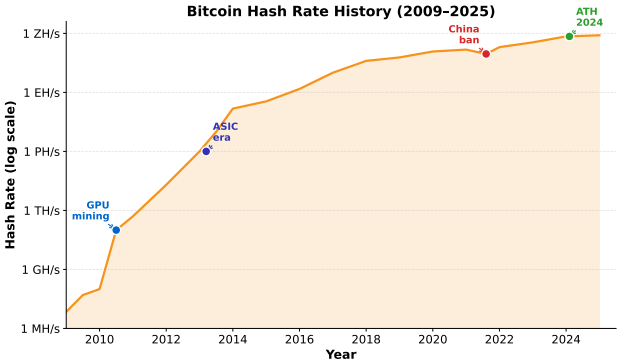
Bitcoin's hash rate has grown by a factor of 10^{15} since launch, reflecting the progression from CPUs to purpose-built ASICs.

Hardware eras:

1. **2009–2010:** CPU mining (MH/s)
2. **2010–2013:** GPU mining (GH/s)
3. **2013–2016:** Early ASICs (TH/s)
4. **2016–2021:** Industrial ASICs (EH/s)
5. **2021–present:** Institutional scale (ZH/s)

Key events:

- China mining ban (Jun 2021): −50% hash rate
- Recovery in 6 months as miners relocated
- 2024: hash rate exceeds 1 ZH/s for first time



- **What you see:** Log-scale hash rate timeline from 2009 to present with key events annotated.
- **Key pattern:** Exponential growth interrupted only by major regulatory shocks (China ban) – and rapid recovery.
- **Takeaway:** The network's security (measured by hash rate) has never been higher, making 51% attacks increasingly impractical.

Mempool Dynamics: The Waiting Room

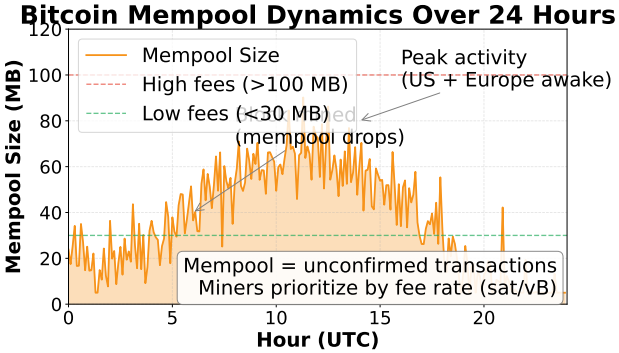
The **mempool** (memory pool) is each node's local collection of unconfirmed transactions waiting to be included in a block.

How it works:

- User broadcasts a transaction to the network
- Nodes validate and add it to their mempool
- Miners select transactions sorted by **fee rate** (sat/vB)
- When a block is mined, included transactions leave the mempool

Congestion patterns:

- Normal: 5,000–20,000 pending transactions
- Congested: 100,000+ (e.g., Ordinals inscription waves, 2023)
- During congestion, low-fee transactions can wait hours or days



- **What you see:** Mempool size over time, showing fee-rate bands stacking up during congestion.
- **Key pattern:** High-fee transactions clear quickly; low-fee transactions accumulate during busy periods.
- **Takeaway:** The mempool is a real-time auction – understanding it helps you set appropriate fees.

Mempool = unconfirmed transactions waiting to be mined. Each node has its own mempool – there is no single global one.

Fee Market: How Transaction Priority Works

Bitcoin's fee market is a **first-price auction** where users bid for limited block space by attaching fees to their transactions.

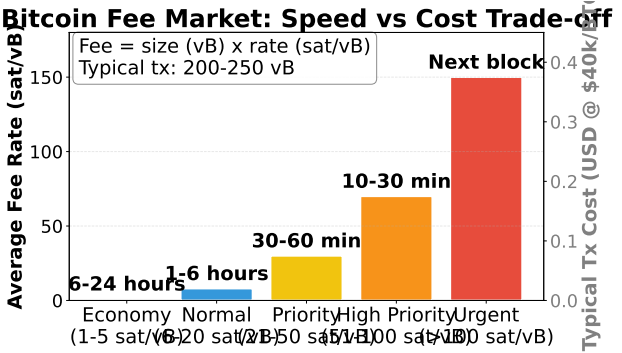
How miners select transactions:

1. Sort mempool by fee rate (sat/vB)
2. Fill block with highest-paying transactions first
3. Stop when block is full (4 MW / ∼1.5–2 MB)

Fee estimation tools:

- `mempool.space` – real-time fee estimates
- `blockchain.info` – historical fee data
- Wallet built-in estimators

RBF (Replace-By-Fee): If your transaction is stuck, you can rebroadcast it with a higher fee. The new version replaces the old one in miners' mempools.



- **What you see:** Fee rate distribution showing how miners prioritize transactions by sat/vB.
- **Key pattern:** Fee rates spike during demand surges and collapse when blocks have spare capacity.
- **Takeaway:** Fee rate (sat/vB) determines priority, not the absolute fee amount. A compact SegWit transaction saves money.

SegWit and transaction batching reduce effective size, lowering fees. Smart wallets optimize UTXO consolidation.

Block Propagation and Network Topology

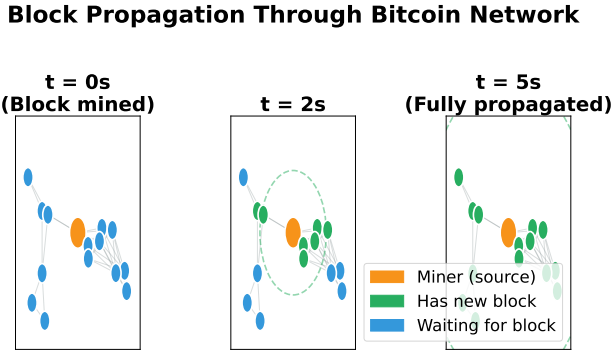
When a miner finds a valid block, it must reach the rest of the network quickly. Slow propagation wastes hash power and increases orphan risk.

Propagation mechanisms:

- **Compact blocks** (BIP 152): Send only short transaction IDs; receiving nodes reconstruct from their mempool
- **FIBRE:** Low-latency relay network used by miners
- **Standard P2P:** Gossip protocol – each node relays to 8+ peers

Network nodes (2025):

- Full nodes: ∼20–25K reachable (∼600 GB)
- Mining pools: ∼15 major pools
- SPV clients: millions (mobile wallets)
- Total: ∼50,000+ (many behind NAT)



- **What you see:** Block propagation time distribution across the Bitcoin network.
- **Key pattern:** Most nodes receive new blocks within seconds – fast enough to minimize orphan rates.
- **Takeaway:** Anyone can run a full node on a Raspberry Pi + 1 TB SSD – consensus is enforced by validators, not just miners.

Blocks propagate globally in 2–5 seconds via P2P gossip. Geographic leaders: US, Germany, Canada

Chain Selection: Nakamoto Consensus

When two miners find valid blocks at nearly the same time, the network temporarily has two competing chain tips. Nakamoto consensus resolves this elegantly.

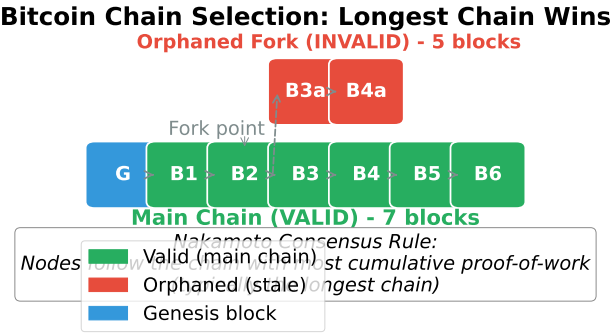
The rule: Follow the chain with the **most cumulative proof-of-work** (often simplified as "longest chain").

How resolution works:

1. Two valid blocks A and B found at similar times
2. Nodes build on whichever they received first
3. Next block extends one chain, making it longer
4. All nodes switch to the longer chain; shorter becomes an orphan

Confirmation security:

Confirms	Risk	Use Case
0	Very high	Never for value
1	High	Small purchases
3	Low	Standard commerce
6	Very low	Large transfers



- **What you see:** A chain fork with two competing tips resolving when one chain accumulates more work.
- **Key pattern:** Forks are normal and resolve quickly – usually within one additional block.
- **Takeaway:** "Longest chain wins" is the simplest consensus rule, but waiting for confirmations reduces reversal risk exponentially.

"6 confirmations" became the standard based on Satoshi's original analysis in the Bitcoin whitepaper (Section 11).

SegWit and Taproot: How Bitcoin Upgrades

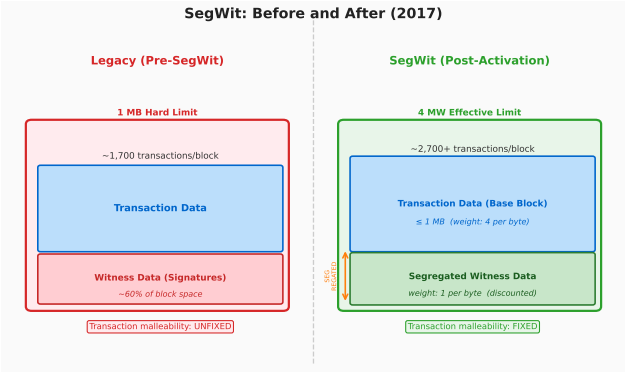
Segregated Witness (August 2017) was Bitcoin's most significant upgrade – a soft fork that restructured transaction data.

What SegWit changed:

- Witness data moved **outside the main block**
- Block limit: **4 MW** (was 1 MB)
- Fixes **transaction malleability**
- Enables **Lightning Network**
- ∼40% more transactions per block

Taproot (November 2021):

- **Schnorr signatures:** More efficient than ECDSA, enable key aggregation (multiple signers produce one signature)
- **MAST:** Merkelized Abstract Syntax Trees – only reveal the spending condition actually used
- **Privacy:** All Taproot transactions look identical on-chain



- **What you see:** Before/after block structure with witness data segregated.
- **Key pattern:** Signature bytes cost 1/4 of non-witness bytes in weight calculation.
- **Takeaway:** Soft fork – old nodes still validate. SegWit now used in ∼90% of transactions.

Lightning Network: Scaling Bitcoin

Lightning Network: Bitcoin's Layer 2 scaling solution, enabled by SegWit's fix of transaction malleability.

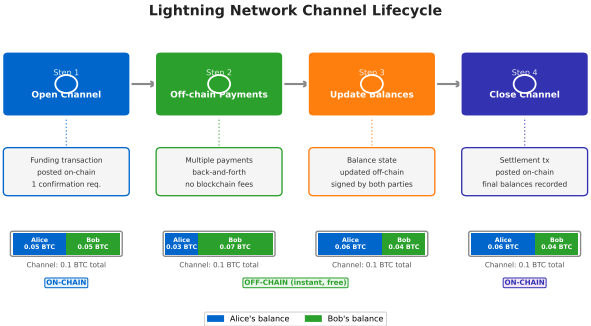
How it works:

1. Open a **payment channel** with an on-chain transaction
2. Exchange **unlimited off-chain payments** instantly
3. Close the channel with a final on-chain settlement
4. Payments can route through intermediary nodes

Current capacity (2025):

- 5,000+ BTC across 15,000+ nodes
- 60,000+ payment channels
- Millions of payments monthly

Key benefit: Only two on-chain transactions needed (open + close), regardless of how many payments flow through the channel. Enables micropayments (sub-cent) and instant settlement.



- **What you see:** The lifecycle of a Lightning payment channel from opening to closure.
- **Key pattern:** Only two on-chain transactions needed (open + close), regardless of how many payments flow through the channel.
- **Takeaway:** Lightning enables micropayments and instant settlement – solving Bitcoin's throughput limitation.

Lightning processes millions of payments monthly, primarily in El Salvador and Africa. Adoption growing steadily.

Fork History and Bitcoin's Future

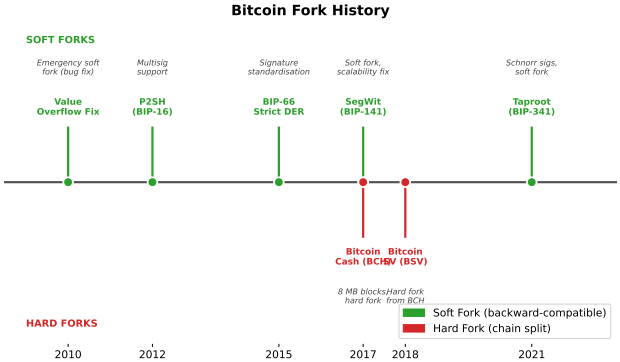
Three types of forks:

1. **Temporary fork:** Two miners find blocks simultaneously. Resolves naturally within 1–2 blocks. Harmless.
2. **Soft fork:** Backward-compatible rule *tightening*. Old nodes still accept new blocks. Example: SegWit (2017).
3. **Hard fork:** Non-backward-compatible rule change. Creates a permanent chain split if not all nodes upgrade.

Notable hard forks:

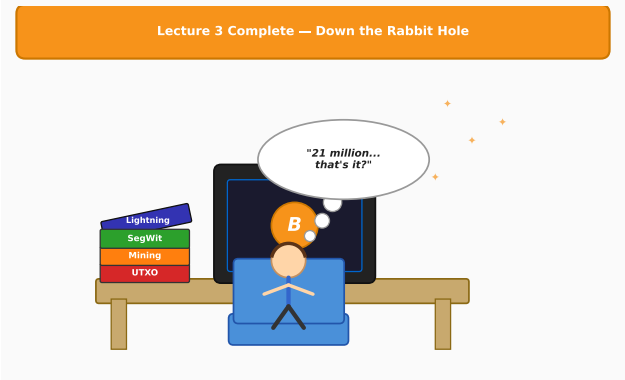
- **Bitcoin Cash** (Aug 2017): 8 MB blocks
- **Bitcoin SV** (Nov 2018): 128 MB blocks
- **Bitcoin Gold** (Oct 2017): GPU-friendly mining

None achieved Bitcoin's network effect or market cap. Bitcoin's upgrade process is deliberately conservative – social consensus matters as much as code.



- **What you see:** Timeline of major Bitcoin forks and protocol upgrades from 2009 to present.
- **Key pattern:** Soft forks (SegWit, Taproot) succeed; hard forks create splinter chains that lose relevance.
- **Takeaway:** Bitcoin's governance is its immune system – contentious changes get rejected by the network.

One Last Thought. . .



Now you understand Bitcoin's machinery – UTXOs, scripts, mining, and the fee market. A few straightforward rules, applied by thousands of independent nodes, secure over $1 trillion in value with zero downtime since 2009.

Next lesson: *L04: Consensus Mechanisms* – PoW vs PoS, Nakamoto vs BFT, and the trade-offs that define every blockchain.

The best way to understand Bitcoin is to use it: send a small transaction and watch it propagate on mempool.space.

Key Takeaways

1. **UTXO model:** Track individual outputs, not balances. Transactions consume UTXOs and create new ones. Your "balance" is the sum of UTXOs you can unlock.
2. **Block structure:** 80-byte header (6 fields) commits to transactions via Merkle root. Only the header is hashed for PoW.
3. **Mining:** SHA256 lottery with difficulty adjustment every 2,016 blocks. Self-regulates to 10-minute blocks regardless of hash rate changes.
4. **Supply:** Hard-capped at 21M BTC via halving schedule. Over 93% already mined. No central authority can change the rules.
5. **Fee market:** Users bid for limited block space. Fee rate (sat/vB) determines inclusion order. RBF and CPFP allow fee adjustment after broadcast.
6. **Chain selection:** Most cumulative proof-of-work wins. Confirmations reduce reversal risk exponentially. Six confirmations is the gold standard for large transfers.

Review question: Calculate the difficulty adjustment if 2,016 blocks took 10 days instead of 14. What happens to block time?

Summary / Next Lesson Preview

Core tension resolved: Bitcoin replaces institutional trust with cryptographic proof – but trustless money means no password resets, no customer support, and no reversals. The trade-off is real, and understanding the technical machinery helps you evaluate whether it is worth making.

Key Vocabulary:

- **UTXO** – Unspent Transaction Output
- **Coinbase tx** – First tx in block; creates new BTC
- **Nonce** – 4-byte number miners vary to find valid hash
- **Merkle root** – Commitment to all txs in the block
- **Difficulty target** – Threshold for valid hashes
- **Halving** – Block reward halved every 210,000 blocks
- **SegWit** – Witness data separated; fixes malleability
- **Taproot** – Schnorr + MAST; privacy + efficiency
- **P2PKH / P2TR** – Script output types (legacy / Taproot)
- **RBF / CPFP** – Fee-bumping mechanisms
- **Soft fork** – Backward-compatible protocol upgrade
- **Hard fork** – Incompatible split; creates new chain

Next lesson: *L04: Consensus Mechanisms* – Nakamoto Consensus, the longest-chain rule, 51% attacks, selfish mining, and how Bitcoin's hash rate makes rewriting history economically unfeasible.

Bitcoin white paper: Nakamoto (2008). Full node software: Bitcoin Core (bitcoincore.org). Block explorer: mempool.space.