7

Blockchain - Next Generation

his chapter explores the evolution of blockchain technologies beyond the foundational models of Bitcoin and Ethereum, focusing on advancements that address energy consumption, scalability, and interoperability. You will learn how alternative consensus mechanisms, such as Proof-of-Stake, Proof-of-Burn, and Proof-of-Authority, aim to mitigate the inefficiencies of Proof-of-Work while introducing new trade-offs in terms of security and decentralization. The chapter explains how Layer-One solutions, such as DAG-based systems and sidechains, reimagine the structure of transaction processing, while Layer-Two approaches, including payment channels and rollups, improve throughput by offloading activity from the main chain. You will also gain an understanding of interchain technologies like Polkadot and Cosmos, which enable different blockchains to communicate and operate within a shared ecosystem. The role of bridges in enabling asset and data transfers across chains is examined with attention to their security risks and design trade-offs. Finally, the chapter revisits Namecoin as an early and ambitious attempt to apply blockchain to domain name services, illustrating the challenges of adoption and long-term utility in decentralized infrastructure.

Bitcoin introduced a decentralized digital currency, and Ethereum expanded blockchain use through smart contracts. Over time, these platforms have not only evolved technically but also given rise to new tools and application domains such as decentralized finance and digital asset ecosystems. This chapter examines major advancements in core blockchain technology, beginning with recent updates in fundamental blockchain consensus mechanisms.

7.1 Alternative Consensus Mechanisms: Proof-of-X

Blockchain has faced increasing criticism due to several design choices, particularly concerning its energy consumption. Proof-of-Work (PoW), the original block-mining algorithm, is often described as solving a mathematical puzzle. In reality, it functions as a lottery where miners must try numerous numbers (nonces), often averaging around 10^{23} attempts in Bitcoin, to find a valid block solution. Only one Bitcoin miner wins this lottery every 10 minutes, and all other computations are wasted. As the blockchain-focused author Tim Swanson notes, this network of computers, or miners, resembles a "peer-to-peer heat machine" due to the vast energy expended.

Since PoW's inception, there have been limited attempts to distribute partial credit to other successful miners (e.g., ommers on Ethereum) or allow multiple miners (e.g., in IOTA). Nevertheless, PoW remained the primary mining algorithm, albeit with significant criticism. For instance, in 2014, it was estimated that producing one bitcoin cost the equivalent of 15.9 gallons of gasoline. [1] This situation has worsened as more miners joined, increasing the environmental impact. The CEO of Filament, Eric Jennings, describes this wasted energy as the cost of having no central authority.

To address these concerns, Proof-of-X alternatives have been developed, aiming to reduce computational inefficiency by modifying assumptions made in PoW. Proof-of-X encompasses various methods requiring miners to demonstrate either work or wealth before block creation.

In early Proof-of-Stake (PoS) designs, wealth was represented by "coin age," with a miner's stake calculated as the number of coins multiplied by their age. Most contemporary PoS systems (including Ethereum's) do not use coin age; instead, they often select validators in proportion to stake weight (and sometimes randomly) without an explicit coin age metric. Once used, a coin's age resets to zero, necessitating an accumulation period before it regains wealth value.

Table 7.1 *Comparison of PoW-based and PoS-based Consensus Protocols*

	Proof-of-Work	Proof-of-Stake
Resource	Computing power	Stake (cryptocurrency)
Access	Permissionless	Permissionless
Energy consumption	High	Low
Fork occurrence	Frequent	Less frequent
Common attacks	Double spending, Self-mining, Eclipse	Nothing-at-stake, Stake grinding, Stake bleeding
Security assumption	Honest majority of hash power	Honest majority of stake
Hardware requirement	Specialized mining rigs	Commodity hardware
Incentives	Block reward, fees	Staking reward, fees

[1] For environmental cost data, see `http://chartalist.org/book/footprint`.

PoS introduces a different approach to mining, reducing energy waste by removing continuous computational work. Furthermore, Proof-of-Burn takes this idea further by requiring miners to send coins to a "verifiably unspendable" address, permanently removing them from circulation. In this scheme, the miner who burns the most coins is selected to mine the next block. Though effective, this reduces the total coin supply over time, so it's only viable when transaction fees or rewards remain high enough. In Table 7.1 we show a comparison of PoW and PoS-based consensus schemes.

In Proof-of-Capacity, memory-based nodes that allocate storage for network functions are chosen as miners, shifting the mining process away from pure computation or wealth destruction.

Despite these alternatives, PoW has a distinct advantage: it discourages malicious behavior through high computational costs. In PoS and burn-based models, miners can duplicate blocks across competing forks without penalty, a problem known as the "nothing at stake" issue. This flaw enables miners to act without repercussion, weakening the security assumption foundational to PoW. Modern PoS implementations include mitigations for the stake problem. For example, Ethereum's PoS slashes (penalizes) validators who sign conflicting checkpoints or blocks on different forks, disincentivizing such behavior.

The philosophical basis for PoW is the assumption that all network participants could act maliciously, warranting high computational costs to deter attacks. However, in many networks, where participants are largely cooperative or identifiable, this assumption is often too conservative, making PoW's energy costs disproportionately high.

Committee-based consensus protocols rely on a fixed or rotating group of known validators who are authorized to confirm transactions on behalf of the network. Unlike proof-based systems, where influence derives from scarce resources such as computing power or stake, committee-based systems operate under partial trust: participants are identified, vetted, or elected, and their agreement is sufficient to finalize blocks. These protocols, exemplified by Practical Byzantine Fault Tolerance (PBFT), HotStuff, and Tendermint, achieve deterministic consensus with high throughput and short confirmation latency. Their design is particularly suited to private or consortium blockchains, where membership is controlled and efficiency outweighs the need for complete openness. The main differences between proof-based and committee-based approaches are summarized in Table 7.2.

Recognizing the limitations of pure proof-of-work/stake or committee-based designs, blockchains have introduced Delegated Proof of Stake (DPoS) as a hybrid approach. In DPoS, stakeholders vote to elect a small group of delegates who validate blocks, combining stake-based incentives with committee-

Table 7.2 *Comparison of Proof-based and Committee-based Consensus Protocols*

	Proof-based	Committee-based
Example algorithm	PoW, PoS	PBFT, HotStuff
Use cases	Public blockchains	Private or consortium chains
Access	Open (anytime)	Conditional
Authorization	No	Yes
Confirmation latency	Long	Short
Decentralization	High	Low
Deterministic consensus	No	Yes (most cases)
Fault tolerance	Probabilistic (majority honest)	Byzantine ($\leq$ 1/3 faulty)
Scalability	Limited	Higher
Transaction anonymity	Strong	Weak
Throughput	Low	High
Energy efficiency	Low	High

style finality mechanisms inspired by PBFT. This structure improves scalability and reduces confirmation latency compared to PoW and PoS. However, while DPoS aims to preserve decentralization through open participation in delegate elections, in practice, it faces risks of vote-buying, cartel formation, and collusion, raising concerns about the long-term integrity of this consensus model.

As we explore various Proof-of-X mechanisms, it becomes clear that sustainability and energy efficiency remain pivotal challenges in blockchain design. The next section examines the energy-efficient and environmentally sustainable blockchain solutions.

7.1.1 Energy Efficiency and Sustainability

One of the primary motivations behind the development of alternative consensus mechanisms is the need for more energy-efficient and sustainable blockchain networks. Proof of Authority (PoA) is one such mechanism designed for private or consortium blockchains where validators are pre-approved, trusted entities. Unlike PoW and PoS, PoA does not require extensive computational resources, as the validation process is handled by a small group of known participants. This leads to significantly lower energy consumption while maintaining high throughput and low transaction latency. VeChain and the POA Network are two notable examples of platforms utilizing PoA for applications such as supply chain management, where the trust model is based on the authority of known validators rather than anonymous miner competition.

Another innovative approach is Proof of Burn (PoB), which requires validators to burn (permanently destroy) a portion of their tokens to demonstrate their

commitment to the network. Here, burning refers to sending coins to an address whose private keys cannot be used (computed). For example, on Ethereum, the null address `0x00...00` is such an address that is used to burn tokens and ETH. This burning mechanism reduces the circulating supply of tokens, potentially increasing their value over time while avoiding the energy consumption associated with PoW. Slimcoin, a blockchain utilizing PoB, shows how this model can promote long-term stability and sustainability without relying on energy-intensive mining.

In addition to PoA and PoB, several other consensus models have emerged to optimize specific aspects of blockchain performance:

- *Proof of Stake-Time (PoST)* combines token staking with a temporal component, requiring validators to commit their stake for a fixed duration. This design encourages long-term participation and network stability by tying consensus power to both capital and time commitment.
- *Proof of History (PoH)*, introduced by Solana, serves as a cryptographic timestamping system that establishes a verifiable sequence of events. When integrated with Proof of Stake, PoH enables high transaction throughput by reducing coordination overhead between validators.
- *Proof of Replication (PoRep)* and *Proof of Space-Time* are utilized in decentralized storage networks such as Filecoin. These protocols require miners to demonstrate that they are storing unique data over a specified period.
- *Proof of Elapsed Time (PoET)* leverages trusted execution environments (e.g., Intel SGX) to generate randomly timed delays, ensuring fairness in leader selection. Used in platforms like Hyperledger Sawtooth, PoET relies on hardware-based attestation rather than competition.

To address the limitations of individual consensus mechanisms, many blockchain networks are now adopting hybrid models that combine the strengths of multiple consensus strategies. Hybrid consensus models aim to balance the trade-offs between security, decentralization, and efficiency. For instance, a blockchain might combine Proof of Stake with Proof of Authority to create a system that benefits from the scalability and energy efficiency of PoA while maintaining the security and decentralization of PoS.

While Proof-of-X mechanisms improve energy efficiency, they do not by themselves resolve blockchain scalability. To meet rising transaction volumes and application complexity, blockchains rely on dedicated scalability solutions, both on-chain and off-chain. The following sections examine approaches that aim to increase throughput while maintaining decentralization and security.

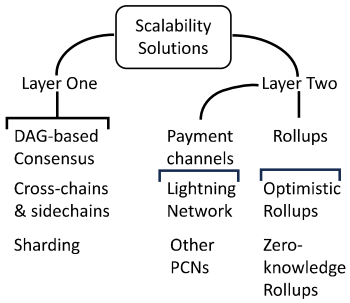

Figure 7.1 Solutions addressing the scalability side of the blockchain trilemma, involving on-chain (Layer One) and off-chain (Layer Two) strategies. On-chain solutions, like sharding, work to divide the blockchain state into smaller pieces for parallel processing, improving scalability without fully compromising security and decentralization. Off-chain solutions, such as sidechains and state channels (e.g., the Lightning Network and Raiden), enable transactions to occur off the main chain, reducing the load on the blockchain while maintaining its security guarantees. The figure is inspired by [327].

7.2 Scalability Solutions for Blockchain

A blockchain grows in time as new blocks are added, but the completed transactions are not removed. Blockchain platforms, such as Ethereum, refund contract and value deletion operations to free memory, but this does not reduce what is already stored in blocks on disk. The transaction history is never reordered, removed, or summarized, and this is taking a toll on resources. Blockchain storage sizes have been steadily increasing (around 1TB for Bitcoin). Ethereum software, such as Geth or Parity, uses pruning to keep the latest blocks in memory only, and most blockchain software can be configured to store block headers only without transactions. However, such a limited view on a blockchain requires trusting header data to be correct, which creates data security risks.

At the heart of blockchain scalability lies the scalability trilemma (Figure 7.2), which posits that achieving scalability, security, and decentralization simultaneously is exceptionally challenging. Enhancing one aspect often compromises the others. For example, increasing transaction throughput might lead to reduced decentralization if only a few nodes can handle the increased load, thereby making the network more susceptible to attacks. Decentralization has been studied in developing new consensus mechanisms such as PoS and DPoS. Security improvements have been implemented with Taproot and SegWit. For the last aspect, scalability, we list some improvements in Figure 7.1.

Scalability in blockchains has been explored mainly through Layer 1 and

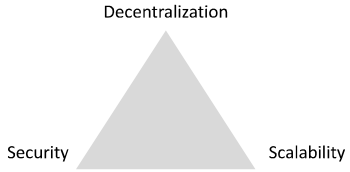

Figure 7.2 The blockchain trilemma illustrating the challenge of achieving decentralization, scalability, and security simultaneously in blockchain systems. Blockchains typically prioritize two of these properties, often compromising the third. For example, Bitcoin emphasizes decentralization and security at the cost of scalability, while other blockchains may enhance scalability but reduce decentralization.

Layer 2 solutions. As representative approaches, we begin with sidechains, which illustrate off-chain extensions of the base protocol, and sharding, which exemplifies on-chain partitioning of state and computation.

7.2.1 Layer-One Protocols

Layer-One Protocols also include alternative architectures beyond traditional block-based structures. One notable example is the Directed Acyclic Graph (DAG), which confirms transactions individually rather than bundling them into blocks. A second alternative involves using parallel chains. In this section, we will cover these approaches.

Directed Acyclic Graphs: An Alternative Layer-One Architecture

Unlike traditional blockchain systems that bundle transactions into blocks and rely on block mining, a Directed Acyclic Graph-based system confirms transactions individually. In other words, the confirmation process is distributed across all participants. The approach has been popularized by the IOTA blockchain in its Tangle graph.

Each DAG user validates prior transactions when submitting new ones, which eliminates the need for blocks and enables parallel transaction validation. This structure makes the system highly suitable for environments with limited computational resources, such as IoT devices, and creates a more scalable and efficient network.

Each DAG transaction confirms two or more previous transactions, referred to as *tips*, and selecting these tips is central to the system's security. Typically, the tip selection process is approximate because the DAG contains too many transactions to efficiently identify the optimal tip. For instance, in IOTA, the

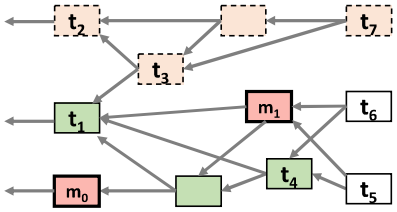

Figure 7.3 IOTA 1.0 Tangle with milestones m_0, m_1. Edges indicate approvals, not spending. For example, t_5 approves both t_4 and m_1. Transactions t_1 and t_2 conflict by spending the same input (not shown here). Under m_0, the past cone does not include t_1, so t_1 is not confirmed. Once a later milestone m_1 references the branch containing t_1 (so $t_1 \in \text{Past}(m_1)$), any transaction such as t_3 that approves the conflicting t_2 lies on an invalid branch and should not be selected as a tip. We show such invalid transactions with a dashed shape. t_5 and t_6 are not yet confirmed by a milestone transaction (but they appear valid).

Monte Carlo Markov Chain algorithm is used to probabilistically select tips based on factors like transaction age or weight, balancing performance and security. Once the tips are selected, the new transaction is added to the DAG and confirmed by subsequent ones (Figure 7.3).

However, confirming individual transactions does not entirely prevent double-spending attacks. To address this issue, some DAG-based systems use decentralized methods to validate and secure transactions without relying on a central authority. In IOTA 2.0, for example, consensus is achieved through a distributed process where nodes validate transactions based on a reputation system (called Mana) and a voting mechanism to resolve conflicts like double-spending. This decentralized validation ensures the integrity of the network by confirming transactions through the agreement of multiple nodes. As a result, the system no longer requires a central entity.

The transaction model in a DAG-based system typically includes two types of transactions: value transactions and zero-value transactions. Zero-value transactions store non-monetary data, such as sensor readings, messages, or fragments of larger transaction signatures. They do not transfer value but play an essential role in enabling data storage and integrity within the DAG structure. Value transactions transfer coins. IOTA 1.0 followed the Unspent Transaction Output (UTXO) model of Bitcoin, where each transaction consumes one or more inputs and generates one or more outputs. The transaction model in IOTA 2.0 has moved away from the UTXO model. The newer version of IOTA, particularly with the Stardust update, introduces a new account-based model instead of UTXO. This means that instead of tracking unspent transaction out-

puts for each address, the system maintains balances directly at the account level.

As transactions accumulate in the DAG, its size can increase quickly. To control this growth, DAG-based systems periodically perform snapshots, which clear the transaction history while keeping the current state of account balances intact. Only specialized nodes, called permanodes, retain the entire transaction history. Regular nodes only store the most recent transaction data to minimize storage and computational overhead.

Feature	DAG-based System	Block-based System
Structure	No blocks	Blocks
Validation	Parallel	Sequential
Scalability	High	Low
Consensus	Leaderless	PoW/PoS
Throughput	High	Typically low
Energy Efficiency	High	Typically low
Confirmation Speed	Fast	Typically slow
Centralization Risk	Low	High
Transaction Security	Moderate	High

Table 7.3 *Comparison between DAG-based systems and traditional block-based systems. The table highlights structural and performance differences in typical implementations. Values such as throughput, energy efficiency, and confirmation speed are often lower in blockchains that rely on Proof-of-Work, but newer designs (e.g., Ethereum's Proof-of-Stake) demonstrate that block-based systems need not be constrained to these limitations.*

To appreciate the scalability and energy efficiency trade-offs between different architectures, Table 7.3 compares DAG-based and block-based systems, highlighting the distinct advantages and limitations of each approach. At first glance, the table shows a favorable view of DAG. However, it is important to note that in DAG-based systems like IOTA 2.0, transactions are not as inherently secure as in block-based systems because there is no global view or unified structure that contains all transactions. The decentralized, tip-based validation approach can lead to temporary inconsistencies or delays in reaching consensus, as different parts of the network may have conflicting views of the transaction history until enough confirmations propagate through the system. As a result, while DAG systems offer scalability and efficiency, they may sacrifice some degree of transaction security compared to the global finality achieved in block-based systems.

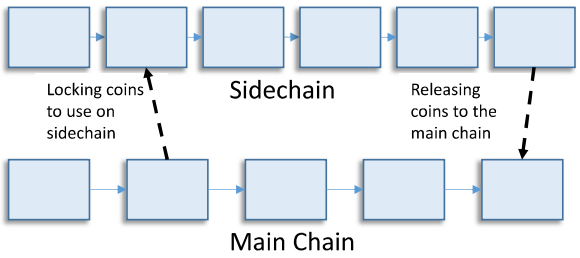

Figure 7.4 Illustration of a main blockchain and its sidechain. The main chain processes the primary set of blocks, while the sidechain enables additional functionality. The first transaction represents the transfer of assets from the main chain to the sidechain, while the second transaction demonstrates the return of assets from the sidechain back to the main chain.

Sidechains

Sidechains are parallel blockchains connected to a main blockchain (also called the parent or main chain) that allow for the movement of assets or tokens between the two chains (see Figure 7.4). Sidechains are designed to extend the capabilities of the main blockchain by offloading certain transactions, processing tasks, or applications. This helps to improve scalability, reduce congestion on the main chain, and allow for the experimentation of new features without disrupting the security and integrity of the main blockchain. For example, the main chain could handle the most critical, high-value transactions, while smaller, less crucial transactions are processed on a sidechain. This division of labor reduces congestion and improves the overall throughput of the system.

The main feature of sidechains is the two-way peg, which allows assets to be transferred between the main chain and the sidechain while maintaining their value. This process works as follows:

(i) Assets (such as tokens) are locked on the main chain, typically by sending them to a designated smart contract. This smart contract is designed to securely hold (or "lock") the assets, ensuring they are no longer available on the main chain while in use on the sidechain.
(ii) Once locked, an equivalent amount of assets is unlocked or created on the sidechain.
(iii) When a user wants to return assets to the main chain, the reverse process happens; the assets on the sidechain are locked or burned, and the original assets on the main chain are unlocked.

Sidechains can use different consensus mechanisms compared to the main

chain. For example, if the main chain uses Proof of Work as its consensus protocol, the sidechain might use Proof of Stake. This flexibility allows sidechains to experiment with alternative consensus models or improve performance for specific use cases. However, this independence also brings security risks; sidechain consensus usually relies on weaker mechanisms or fewer validator nodes. The sidechain may be vulnerable to attacks, such as double-spending or 51% attacks. To mitigate this, some systems employ federations or trusted entities to validate transactions between the main chain and the sidechain, ensuring that the transfer of assets is properly managed. Additionally, sidechains may implement advanced security measures, such as cryptographic proofs, to guarantee that transactions are legitimate.

For sidechains to function effectively, there needs to be seamless communication between the main chain and the sidechain. This communication is often achieved through three mechanisms:

(i) Smart Contracts: These smart contracts lock and release assets between the two chains, managing the movement of tokens or data in a secure and automated way.
(ii) Notary Schemes: Trusted third parties, or notaries, can also verify and approve transactions between the chains. However, this adds a level of centralization, which may reduce the decentralization benefits of blockchain technology.
(iii) Cross-Chain Bridges: These are specialized protocols that enable communication and asset transfer between chains without requiring changes to the core blockchain architecture.

Smart contracts offer low delay but higher setup complexity; notary schemes are simpler but can introduce delays from third-party verification, while cross-chain bridges aim for efficient transfers with moderate implementation complexity. Notary schemes are generally easier to hack due to their reliance on trusted third parties, which introduces a central point of failure. In contrast, smart contracts and cross-chain bridges can be more secure when well-audited, although vulnerabilities in smart contract code or bridge protocols can still pose risks.

We summarize the sidechain discussion in Table 7.4. Sidechains provide an effective way to enhance blockchain networks by increasing scalability, improving transaction speed, and allowing for innovation. They enable blockchains to handle more diverse and complex applications without compromising the integrity or security of the main chain. However, the security of sidechains is a critical factor, and ensuring robust security measures is essential for their widespread adoption.

Aspect	Main Chain	Sidechain
Transaction Speed	Slow	Fast
Transaction Fees	High	Low
Consensus Mechanism	Established (e.g., PoW)	Variable (e.g., PoS, DPoS)
Security	High	Lower (separate consensus)
Decentralization	High	Lower
EVM Compatibility	Optional	Yes
Asset Transfer Process	-	Requires lock and unlock

Table 7.4 *Comparison of main chains and sidechains across various aspects, highlighting the trade-offs between security, scalability, decentralization, and transaction efficiency. Sidechains enhance scalability and reduce fees at the cost of some decentralization and security features compared to main chains.*

7.2.2 Layer-Two Protocols

Layer 2 solutions are designed to address the scalability challenges in blockchain systems by processing transactions off-chain, thereby reducing the load on the main blockchain (Layer 1). These protocols aim to improve transaction throughput, reduce latency, and lower transaction fees. Broadly, Layer 2 solutions are categorized into several types: channels and rollups.

Payment and State Channels

Payment channels are a Layer 2 mechanism that allows two or more participants to transact directly with one another without recording every intermediate transfer on the blockchain. Instead, the blockchain is used only twice: once to open the channel and once to close it. Within the channel, participants exchange signed updates that represent their current balances. Because these updates are off-chain, they can be exchanged rapidly and at negligible cost, while still being enforceable on-chain if needed. Payment channels are thus well-suited for high-frequency, low-value payments. The Bitcoin Lightning Network, launched in 2018, is the most prominent example. A related concept is the *state channel*, which generalizes the idea beyond payments to arbitrary off-chain state updates. In a payment channel, the states are user balances, and updates are made through transactions.

A payment channel has three phases: creation, usage, and closing. In creation, two participants lock funds into a multi-signature (typically a 2-of-2 multi-signature) address and create off-chain transactions by exchanging signed commitment transactions. In the example of Figure 7.5, John and Tom open a

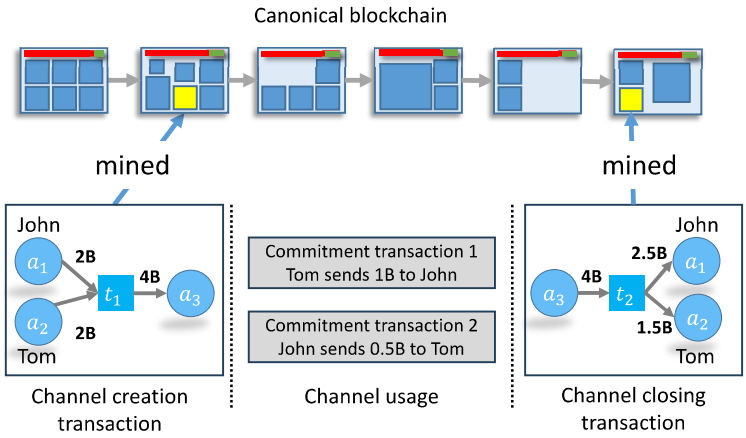

Figure 7.5 An illustration of the payment channel lifecycle between John and Tom (transaction fees are omitted to simplify the example). The diagram shows the channel creation, followed by two commitment transactions where the balances are updated: initially, both hold 2 BTC each. After two commitment transactions, John holds 2.5 units and Tom 1.5 units. The process concludes with the channel closing. Meanwhile, on the top panel, the canonical blockchain moves ahead by creating blocks.

payment channel by depositing 2 Bitcoins each into the channel during the creation phase (channel amounts can be arbitrary). These bitcoins cannot be used in transactions until the channel is closed. Address balances are updated in off-chain "commitment transactions" without involving the blockchain. For instance, in the first commitment transaction, Tom sends 1 BTC to John, who now owns 3 BTC, while Tom holds 1 BTC. In the second commitment transaction, John sends 0.5 BTC to Tom. Once they decide to close the channel, the final balances are settled on-chain, with John receiving 2.5 units and Tom receiving 1.5 units. Throughout this process, only the opening and closing of the channel are published on the blockchain, which significantly reduces on-chain transaction costs and delays.

Payment channels can enable both unidirectional and bidirectional transfers. In unidirectional channels, only one participant can send payments, while the other can only receive funds. However, bidirectional channels, as introduced by [316], have become the standard, allowing both parties to send and receive payments dynamically within the same channel.

While pairwise payment channels can be used for transactions between two participants, they are not sufficient to create a fully functional system where

coins can be transferred across a network of users. To enable broader payments across multiple participants without requiring each user to open a direct channel with every other user, a more complex system is needed. This is where payment channel networks come into play. In a network like the Lightning Network on Bitcoin, coins can be routed through a series of interconnected payment channels. If Tom wants to send coins to Charlie but does not have a direct payment channel with Charlie, the transaction can be routed through Bob, provided Tom has a channel with Bob and Bob has a channel with Charlie. This routing mechanism is necessary to create a payment network out of payment channels between pairwise addresses.

Transitioning from pairwise payment channels to a payment network creates several unique challenges. A user may need to send coins through multiple intermediaries (hops), which requires network discovery. Furthermore, the network changes over time, and some channels use up the channel balance in one way, thereby making the channel unusable in that direction. As a result, a second key challenge is finding the path with the necessary capacity in a short time. If one path alone cannot transfer all coins, the payment must be split into multiple channels. This requires decentralized routing algorithms that can efficiently find paths while preserving privacy.

In a payment channel network, channels charge fees to process transactions as they pass through intermediaries (or "hops") from sender to recipient. Channels can also be marked private to disallow others from using them in the network. Channel fees incentivize channel operators to keep channels open and provide liquidity for routing payments. Each channel owner (node) can set their fee rate, typically defined by two components: a base fee and a fee rate proportional to the amount being transferred. The base fee is a fixed amount for processing any transaction, while the fee rate is a percentage of the transaction amount. When a user initiates a payment, their wallet application calculates the total fees required for each potential path through the network based on the sum of the fees set by each intermediary along the route. Once a route is chosen, the payment proceeds through each channel, and each intermediary deducts its fee from the payment before forwarding it. For example, if a payment passes through three channels, each channel deducts its fee along the way, so the final recipient receives the intended amount minus the cumulative fees. Wallet applications often display the estimated fees before the user confirms the transaction, helping them choose an optimal route that balances low fees with sufficient channel capacity.

To ensure that all parties in the channel act honestly and cannot misappropriate funds, several mechanisms prevent cheating. These mechanisms rely on

time-locked contracts and cryptographic commitments, making fraud attempts detectable and penalizable.

Hashed Time-Locked Contracts (known as HTLCs) are fundamental to most payment channels. They enforce that transactions can only be claimed by revealing a specific hash preimage (a cryptographic "password") within a defined time limit. If the preimage isn't revealed in time, the transaction is canceled, which ensures that funds aren't stolen by one party if another fails to respond.

Revocation keys prevent one party from broadcasting an outdated balance state (cheating) by allowing each party to invalidate previous transactions when both agree on a new balance. By exchanging revocation keys with each updated balance, each party has proof that could invalidate older states. If one party tries to publish an old transaction, the other party can claim all funds in the channel by providing the appropriate revocation key, effectively penalizing the cheater. However, to enforce this, the honest party must be online to detect the fraud attempt.

Watchtowers are third-party services that monitor the blockchain for fraud attempts to publish outdated channel states. If a cheating attempt is detected, the watchtower submits the necessary information to penalize the offender by claiming the funds. For watchtowers to function effectively, they must have an economic reason to monitor the blockchain. This incentive is built into the Lightning Network's penalty mechanism. When a cheating party publishes an old commitment state, the watchtower can release a pre-signed justice transaction on behalf of the honest participant. This transaction redistributes the channel funds so that the dishonest party loses its balance. A small portion of these recovered funds is awarded to the watchtower as compensation for its service, while the rest returns to the honest user. In addition, some proposals envision users paying watchtowers directly through service fees or subscriptions.

Time-Locked Refunds offer protection if one party becomes unresponsive or uncooperative, ensuring the other party's funds are returned. Each channel state has an expiry period, so funds can only be claimed within that time frame. After expiry, funds revert to the sender; this prevents an unresponsive party from holding the funds indefinitely.

Finally, penalization for invalid channel closure discourages malicious actors by allowing the honest party to claim all funds if the other party attempts to close the channel with outdated information. Any attempt to cheat by publishing old states can result in the complete loss of the cheater's funds.

By combining fee structures with mechanisms like HTLCs, revocation keys, watchtowers, time-locked refunds, and penalties for invalid closures, payment channels create a system that is not only efficient but also secure.

Table 7.5 compares payment channels with the main blockchain. Payment

Aspect	Payment Channels	Main Chain
Transaction Speed	Fast	Slow
Transaction Fees	Low	High
Scalability	High	Limited
Finality	Instant (within channel)	Probabilistic
Transaction Privacy	✓	×
Security Dependency	Relies on channel mechanisms	Full blockchain security
Decentralization	Lower	High
Suitable for Microtransactions	✓	×
Fee Flexibility	Customizable	Standardized
Trust Requirements	Requires counterparty trust or watchtowers	Trustless

Table 7.5 *Comparison between payment channels and main blockchain for various transaction and security aspects.*

channels offer enhanced privacy compared to traditional on-chain transactions because most of the transaction activity occurs off-chain, reducing the visibility of individual payments to third parties. Only the opening and closing of the channel are recorded on-chain, which significantly limits the amount of information available publicly. Additionally, because the intermediate updates and balance changes are not visible on the blockchain, external observers cannot trace the frequency, amount, or direction of payments made between the participants. This privacy feature is particularly valuable in cases where users seek to conduct frequent, low-cost transactions (known as microtransactions) without exposing their financial activity to the public. However, some information, like the channel's existence and the final settlement transaction, is still publicly recorded when the channel is closed.

Transaction Rollups

Another Layer 2 solution that has gained traction in recent years is transaction rollups (or simply rollups). Rollups were initially proposed to scale blockchain systems by aggregating and processing transactions off-chain, while the term "rollup" refers to bundling transaction data into batches before publishing them on the blockchain. Unlike pure Layer 2 protocols, which submit multiple off-chain operations in a single on-chain transaction, rollups are often considered hybrid Layer 1/Layer 2 protocols because they publish key information about each transaction on-chain. Figure 7.6 shows states and their stored information on Layer 1.

The deployment of rollups requires a smart contract on the Layer 1 blockchain to handle state management (e.g., storing the Merkle root of the rollup's states), transaction verification, and dispute resolution. This complexity makes rollups less practical for blockchain systems without Turing-complete smart contracts, such as Bitcoin and its derivatives.

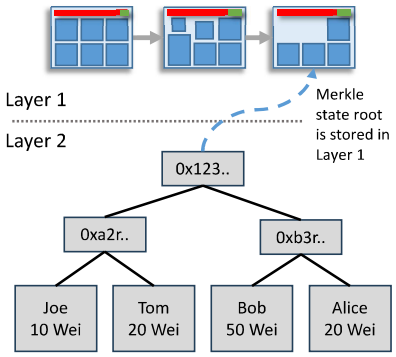

Figure 7.6 Rollups as a Layer 2 scaling solution. Transactions are executed off-chain in Layer 2, where user balances (e.g., Joe, Tom, Bob, Alice) are maintained in a Merkle tree. The Merkle state root summarizing these balances is periodically committed to Layer 1, allowing Layer 1 to serve as the ultimate source of security and data availability. This design reduces transaction costs and increases throughput while preserving the security guarantees of the base blockchain.

The rollup smart contract on the blockchain acts as the root of trust, resolving disputes and initializing the off-chain layer. Users who wish to perform off-chain transactions must allocate funds (e.g., tokens) through this smart contract. The contract stores the rollup's state, typically represented as a Merkle root (e.g., of users' account balances). To update the state, an entity called an aggregator batches, or "rolls up", off-chain transactions and submits them to the smart contract as `calldata` or in blobs (`calldata`, which we covered in Chapter 3.16, is the data passed to a smart contract to execute specific functions). The aggregator compresses the data to reduce congestion, omitting redundant information, such as full addresses and signatures, by using indexed references instead. The smart contract verifies the new Merkle root and updates the rollup's state accordingly.

While the contracts governing the rollup protocol run on Ethereum, the actual computation and state storage occur on a separate virtual machine that is distinct from the Ethereum Virtual Machine. This off-chain virtual machine handles the applications and state transitions, serving as the "Layer 2" of the rollup system. Rollups create their own blocks and maintain their own execution environments, but they do not run consensus protocols in the traditional sense. Instead, they rely on a sequencer to order transactions, while delegating data availability and security enforcement to Ethereum. Because rollups operate off-chain but post critical data and proofs to Ethereum, they are often referred to as hybrid Layer 2 solutions. They function as separate execu-

tion layers, but their correctness and security ultimately depend on Ethereum's Layer 1. Ethereum guarantees the correctness of off-chain computations and ensures the availability of the data used in these computations, making rollups, such as Optimistic Rollups, more secure than fully off-chain scaling solutions like sidechains, which do not rely on Ethereum's security model.

Because rollup batches omit some information and can be submitted by anyone, there must be mechanisms to verify the accuracy of the data submitted. Rollups are broadly divided into two categories based on their approach to verifying state changes: Optimistic Rollups and Zero-Knowledge (zk) Rollups.

Optimistic Rollups

Optimistic Rollups are so named because they operate on the assumption (or optimism) that all transactions in a batch are valid by default. There are four main roles for the Optimistic Rollup users: users, aggregators, validators, and sequencers.

Users deposit Ethereum ERC20 tokens or other supported assets into the rollup's bridge contract on Ethereum's Layer 1. This contract transfers the assets to Layer 2, where the same amount is created and sent to the user's address on the optimistic rollup. Users also initiate transactions in the optimistic rollup system. They submit their transactions to be processed by the network, but they do not directly handle the off-chain execution. Instead, their transactions are sent to operators or validators, who manage the computation and state changes.

Aggregators are nodes responsible for processing the transactions in the optimistic rollup. They aggregate multiple transactions, compress the associated data, and publish this compressed batch to the Ethereum main chain (Layer 1). Aggregators ensure that the off-chain computations are correctly batched and presented on-chain. Before producing blocks, aggregators must provide a bond (collateral) on the Layer 1, similar to a proof-of-stake mechanism. This bond can be slashed (confiscated) if the aggregator posts an invalid block or builds on a previous invalid block, even if their own block is valid. This ensures that operators act honestly.

Sequencers are responsible for ordering incoming transactions, batching them efficiently, and submitting the resulting rollup data to Layer 1.

Validators are tasked with verifying the sequencer's proposed state changes. They execute the submitted transactions using their own copy of the rollup's state. If they find that the final state proposed by the aggregator does not match their computed state, they can initiate a challenge by submitting a "fraud-proof". This fraud proof provides cryptographic evidence of the invalidity of

the operator's state, ensuring the system's correctness through decentralized verification.

In most Optimistic Rollup designs, the terms aggregator and sequencer refer to the same role, since the party that batches user transactions is also responsible for ordering and posting them to Layer 1.

The transactions are not immediately verified on-chain. Instead, after a batch of transactions is processed off-chain, the resulting compressed data (including state roots and transaction details) is posted to the main blockchain. The rollup assumes that all transactions in the batch are valid; hence, no immediate validation or proof is provided.

A key feature of Optimistic Rollups is the dispute resolution process. When the rollup posts the batch to Layer 1, a "challenge period" (typically ranging from hours to a few days) is opened, allowing anyone to review the transactions. If a validator or a participant suspects that a transaction in the batch is fraudulent, they can submit a fraud proof. The fraud proof is a cryptographic challenge that forces the computation of the transaction to prove whether it is valid or invalid. If the challenge is successful, the fraudulent party is penalized, and the batch is corrected. In the fraud case, the sequencer responsible for including the incorrectly executed transaction in a block receives a penalty.

One of the downsides of Optimistic Rollups is the delay in finalizing transactions due to the dispute window. Until the challenge period expires, users must wait for their transactions to be considered final. This makes Optimistic Rollups slower in achieving finality compared to zk Rollups. However, Optimistic Rollups are generally more suitable for applications that involve complex smart contract interactions because they are more flexible when executing arbitrary code off-chain. Examples of projects using Optimistic Rollups include Optimism and Arbitrum, which aim to scale decentralized applications by reducing congestion on Layer 1.

Optimistic Rollups can achieve higher throughput; current implementations for Ethereum can process 2000 transactions per second. However, they trade off speed for security by incorporating a challenge period, which introduces delays in transaction finality as it allows time for fraud detection.

Withdrawing assets from an optimistic rollup back to Ethereum is more complex due to the fraud-proof mechanism. When a user initiates a withdrawal from L2 to L1, they must wait through a challenge period, usually around seven days, before the transaction is finalized. However, the actual withdrawal process is straightforward.

Once the withdrawal request is made on the L2 rollup, the assets are burned on L2, and the transaction is included in the next batch. After this batch is submitted to Ethereum, the user can verify their transaction using a Merkle

proof. After the challenge period passes, the transaction is finalized, and the funds can be withdrawn to the Ethereum mainnet.

To avoid waiting the full week for a withdrawal, users can use a liquidity provider (LP). An LP takes over the pending withdrawal and immediately pays the user on L1 for a fee. The LP verifies the withdrawal by checking the rollup's state before paying, ensuring that the transaction will eventually be confirmed. This way, users get faster access to their funds without compromising security. In Decentralized Finance, LPs play an important role, as we discuss in Chapter 8.

Zero-Knowledge (zk) Rollups

Zero-Knowledge (zk) Rollups provide a distinct approach to scaling by ensuring the correctness of each off-chain transaction through cryptographic proofs before submitting state updates to Layer 1. The key feature is the use of zero-knowledge proofs [165], which allow one party to prove to another that a computation was performed correctly without revealing the underlying data.

Like Optimistic Rollups, ZK-rollups bundle transactions into batches and execute them off-chain, thereby reducing the transaction load on Layer 1. However, unlike their optimistic counterparts, ZK-rollups attach a validity proof to each batch, offering cryptographic assurance that all included state transitions are correct. As a result, ZK-rollups do not require a challenge period and can finalize transactions immediately upon proof verification, enabling faster and more reliable withdrawals to Layer 1.

The state of a ZK-rollup, including account balances and other data, is stored in a Merkle tree. Each transaction batch updates this state, and the updated Merkle root is submitted to Layer 1. A smart contract verifies this root using the attached validity proof, ensuring the integrity of the rollup's state.

On Ethereum, ZK-rollups typically store transaction data as calldata, which is not part of Ethereum's active state but is stored permanently in its history. This allows anyone to reconstruct the rollup state independently. To minimize calldata size and associated gas costs, ZK-rollups often employ compression techniques, such as using indexes instead of full addresses.

To update the rollup's state, operators must submit a "validity proof" attesting that the state transition resulting from a batch of off-chain transactions is correct. These proofs can be in the form of ZK-SNARKs [62] or ZK-STARKs [55]. ZK-SNARKs require a trusted setup to generate public parameters, which can pose a risk if the setup is compromised. ZK-STARKs avoid this requirement, enhancing transparency and scalability, but generally produce larger proofs that are more computationally expensive to verify on-chain.

Withdrawals from a ZK-rollup are fast and straightforward. An account

holder on the rollup initiates the process by submitting an exit request. Once the batch containing this request is processed and its validity proof verified on Layer 1, the account holder can generate a Merkle proof of inclusion and present it to the Layer 1 contract. The contract then releases the corresponding funds to the account holder's Layer 1 address.

Comparison of Optimistic and zk Rollups

A common threat for both Optimistic and ZK Rollups is censorship issues. Censorship in rollups can arise when a centralized entity, such as a sequencer or validator, has control over which transactions are included in a batch submitted to the Layer 1 blockchain. In such cases, the operator may choose to exclude or delay specific user transactions, effectively censoring them. This risk is particularly prominent in systems where the operator has significant authority, as they can prioritize certain transactions or refuse to include others. To mitigate this, many rollups, both Optimistic and zk Rollups, include mechanisms that allow users to bypass the sequencer if they believe they are being censored. For example, users can submit their transactions directly to the Layer 1 contract, which ensures that they are eventually included in the rollup, even if the operator refuses to process them.

Table 7.6 compares Rollups across multiple aspects. It is important to note that zk Rollups require generating complex cryptographic proofs (such as zk-SNARKs or zk-STARKs), making them more computationally intensive and potentially costly to implement. On the other hand, Optimistic Rollups are easier to deploy because they do not require the generation of cryptographic proofs for each batch of transactions. Instead, they rely on a challenge mechanism and economic incentives, which reduces the initial computation overhead but introduces delays in finality due to the dispute period. This makes Optimistic Rollups more suited for complex applications, such as decentralized finance protocols. Furthermore, zk-Rollups face limitations in compatibility with the Ethereum Virtual Machine. The machine was not originally designed with zk-proof systems in mind: it supports arbitrary bytecode, dynamic storage, and numerous opcodes that are difficult to represent efficiently inside proof circuits. In contrast, Optimistic Rollups have full virtual machine compatibility, making it easier to integrate with existing Ethereum-based smart contracts.

7.2.3 A Comparison of Layers

This chapter covered how Layer 1 and Layer 2 solutions are increasingly structured to provide security and scalability in the blockchain ecosystem.

Layer 1 (L1) constitutes the foundational blockchain protocol responsible

Feature	zk Rollups	Optimistic Rollups
Rollup Cost	High	Low
Security Model	Cryptographic	Economic Incentives
Latency	Low	High
Best for Complex Smart Contracts	×	✓
Best for High-Throughput and Fast Finality	✓	×
EVM Compatibility	Limited	Full

Table 7.6 *Comparison of zk Rollups and Optimistic Rollups*

for core tasks such as consensus, transaction validation, and data availability. These are blockchains that operate independently and provide the security guarantees upon which other layers rely. Prominent examples include Bitcoin, Ethereum, and Solana.

Layer 2 (L2) protocols are built atop L1s to improve scalability and transaction throughput. By executing transactions off-chain or in parallel, and submitting cryptographic proofs (e.g., fraud or validity proofs) to the L1, these systems achieve higher performance without altering the base layer. Examples include rollups such as Optimism and Arbitrum on Ethereum.

Layer 3 (L3) represents an emerging category of application-specific chains that operate on top of L2 systems. L3s aim to provide tailored environments for decentralized applications, offering enhanced customization, throughput, and interoperability. Use cases include gaming, decentralized finance, and identity systems. Examples include Orbs and XAI Games deployed on Arbitrum Orbit.

Our next topic, bridges, plays a central role in connecting these layers. They enable cross-layer communication, such as between L1 and L2 (e.g., Ethereum to Arbitrum) and between L2 and L3 (e.g., Arbitrum One to Arbitrum Orbit), thereby facilitating scalable and composable decentralized applications.

7.3 Blockchain Interoperability

As individual blockchains and layered architectures (Layers 1, 2, and 3) have matured, the demand for seamless cross-domain communication has become increasingly urgent. At the center of this interoperability challenge lies bridge design: the dominant mechanism for transferring assets and data across chains and layers. While many systems rely on external bridges, some blockchain ecosystems integrate interoperability into their core architecture, enabling native cross-chain communication without relying on third-party bridges. In this

section, we first turn to the structure and security of blockchain bridges. Next, we teach protocol-level interoperability: Cosmos and Polkadot.

7.3.1 Bridges

We categorize bridges along two orthogonal axes: their *functional type* and *operational trust model*. A summary view is shown in Figure 7.7.

The functional type captures how value (e.g., coins, tokens) is transferred and includes asset-based bridges, liquidity networks and hybrid bridges.

In *asset-based bridges* (also known as burn-and-mint models), the bridge locks or burns an asset θ_1 on the source domain b_1 and creates a representative asset θ_2 on the destination domain b_2. These assets are not inherently equivalent; their linkage is established solely through the semantics of the bridge protocol. They include:

- *Externally-Verified Asset Bridges*, which depend on multisigs, notaries, or sidechains to verify lock events (e.g., Ronin, Wormhole, Multichain).
- *Rollup-Native Asset Bridges*, which leverage L1 consensus to verify L2 state transitions using fraud or validity proofs (e.g., Arbitrum, Optimism, Loopring). These are considered the most trustless.

The settlement process on asset-based bridges may be slow and require specialized verification and challenge code. For example, optimistic rollups like Arbitrum or Optimism impose a 7-day challenge period on withdrawals.

In contrast to asset-based bridges, *liquidity network-based bridges*, such as Connext or Across, avoid minting and instead fulfill user requests through liquidity providers who maintain reserves on both domains. LPs are economically incentivized via transfer fees and, in some protocols, additional yield or reward mechanisms. This design enables faster settlement and circumvents the latency of on-chain verification and challenge periods.

The third type, hybrid bridges, support both functional models, often depending on the specific chain pair or asset type. For instance, *Multichain* combines canonical minting with liquidity provisioning.

In the operational trust model, bridges are categorized based on how activity on the source domain is verified. This model includes trusted, trust-minimized, and trustless designs. Accordingly, bridges are commonly referred to as either trusted or trustless, depending on the extent of external trust assumptions they require.

Trusted bridges rely on a central operator or a small, predefined set of parties to manage the locking and minting of assets across chains. For example, when a user sends tokens from Chain A to Chain B using a trusted bridge, the bridge

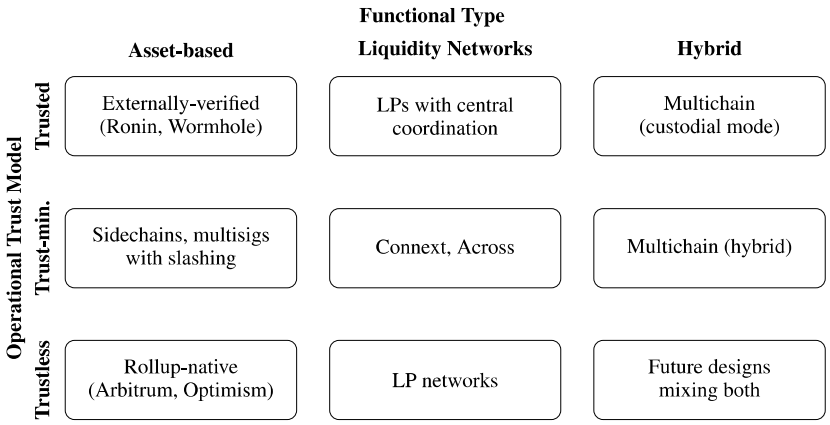

Figure 7.7 Categorization of blockchain bridges along two axes: functional type (asset-based, liquidity network, hybrid) and operational trust model (trusted, trust-minimized, trustless).

operator takes custody of the original tokens on Chain A and issues equivalent wrapped tokens on Chain B. This operator is responsible for holding the locked assets securely and honoring redemption requests. Users must therefore trust that this entity will not mismanage, freeze, or steal the funds and that it will remain operational and solvent.

This custodial setup introduces multiple risks. If the operator is compromised, becomes insolvent, or acts maliciously, users may permanently lose access to their assets. Additionally, the operator can censor withdrawals, delay transactions, or comply with regulatory demands that restrict certain users. While trusted bridges often offer high throughput and support for complex cross-chain functionality, they trade off decentralization and security in favor of convenience and efficiency.

Trustless bridges, on the other hand, operate using smart contracts and algorithms without relying on a central authority. This type makes no new trust assumptions beyond those inherent to the underlying blockchains they connect, making them more secure and decentralized. However, they typically handle simpler transactions and may not support more complex data types or large volumes of transactions as efficiently as trusted bridges.

Relays are a form of trustless bridge that enables one blockchain to verify data or events from another blockchain without relying on a central authority. In a relay system, data from the source chain, such as block headers or specific transaction details, is submitted to the target chain by participants known

as relayers. This data is then validated through smart contracts on the target chain. This ensures the current state of the source chain without requiring trust in a third party. By syncing and validating external blockchain data in a decentralized manner, relays support interoperability without sacrificing decentralization or security.

Several drawbacks hinder relay usage. First, relays often require constant monitoring and updates because the source blockchain's data, such as block headers, must be continuously submitted to the relay smart contract on the target chain. This process requires periodic creation of transactions, which can be costly and resource-intensive, especially for blockchains with high transaction fees like Ethereum. Second, relays introduce some latency, as data from the source chain must be confirmed before being accepted on the target chain. For instance, Bitcoin transactions take around 10 minutes per block confirmation, which can slow down the verification process on Ethereum. Additionally, differences in block finality between chains can make it challenging to accurately sync and verify data.

In general, the design and operation of any type of bridge are not without risks. Issues such as smart contract vulnerabilities, systemic financial risks from wrapped assets, and counterparty risks are significant considerations. For example, bridges that utilize wrapped tokens to represent assets on another blockchain add a layer of risk, as the representation might not always be perfectly secure or accurate. Attackers have exploited these factors to drain bridge assets through complex single-transaction exploits. Bridges, such as Nomad and Ronin, have been exploited by attackers who leverage weak or improperly verified contracts, phishing, and fraudulent transfers. In one case, $624 million was lost through the Ronin bridge, while the Nomad bridge suffered losses of $190 million (see `http://chartalist.org/book/leaderboard` for a current list of bridge exploits). Many bridges have been slow to detect and respond to attacks, sometimes taking days to recognize breaches, which amplifies financial losses. For example, in a 2024 attack, the Ronin team reported suspending bridge operations only 40 minutes after the initial signs of malicious activity.

7.3.2 Protocol-level Interoperability

While most blockchain interoperability relies on external bridges, Polkadot and Cosmos adopt a fundamentally different approach. These platforms are designed with interoperability as a core protocol feature, enabling secure and efficient communication between sovereign chains without relying on third-party bridges.

Polkadot

Polkadot (https://polkadot.com/) is a blockchain protocol designed to connect multiple specialized blockchains into one network. Developed by the Web3 Foundation and led by Dr. Gavin Wood, co-founder of Ethereum, it aims to address the limitations of existing blockchains by providing true interoperability and scalability.

Central to Polkadot's architecture are Parachains, which are individual blockchains that run in parallel within the network. These Parachains are sovereign; they have their own tokens and governance structures, but they benefit from the shared security and interoperability of the Polkadot ecosystem. They connect to the Relay Chain, which handles the network's security, consensus, and cross-chain interoperability.

Polkadot uses the Cross-Chain Message Passing (XCMP) protocol to enable communication between Parachains. XCMP allows for the secure transfer of data and assets between chains within the network. Additionally, Polkadot has bridges to connect with external networks like Ethereum and Bitcoin.

Cosmos and the Cosmos Hub

Cosmos is an ecosystem of interconnected blockchains that aims to create an "Internet of Blockchains". It seeks to solve scalability and interoperability issues by allowing independent blockchains to communicate in a decentralized manner.

The Cosmos Hub is the central blockchain that connects various Zones, which are individual blockchains within the Cosmos network. Each Zone operates independently with its own governance and tokens, but can interact with other Zones through the Hub.

Interchain communication in Cosmos is facilitated by the Inter-Blockchain Communication (IBC) protocol. IBC is a standardized protocol that enables secure and reliable data and asset exchange between heterogeneous blockchains. The network uses the Tendermint consensus algorithm, providing instant finality and high transaction throughput.

7.4 Namecoin: Pioneering Blockchain-Based Domain Services

So far, we have examined next-generation blockchain technologies. Yet one of the earliest experiments with Bitcoin showed how even a basic transaction element could be repurposed into a naming ecosystem. This idea, which began

with a simple log message, led to the creation of Namecoin, a system that still survives today.

Once a block is mined and buried under subsequent blocks (e.g., six confirmations in Bitcoin), it is effectively immutable. Coupled with the decentralized nature of blockchains, this immutability means that once a transaction is mined, every node in the network holds a permanent copy. Such permanence can be used to bypass Internet censorship and disseminate otherwise restricted data. Building on this principle, Namecoin and similar projects used the peer-to-peer and immutable properties of blockchains to offer decentralized domain name services.

Although small in market capitalization, Namecoin holds historical significance as the first Bitcoin fork. Created in 2011, it extended Bitcoin's functionality to store key–value pairs for a decentralized namespace. As its developers describe it, "Namecoin is an experimental open-source technology that improves decentralization, security, censorship resistance, privacy, and speed of certain components of the Internet infrastructure such as Domain Name Services and identities".

A Namecoin record has a key and a value that can be up to 520 bytes in size. As a prominent application of the namespace, Namecoin data blocks store registrations or updates for the .bit online domain names, which are independent of the Internet Corporation for Assigned Names and Numbers (ICANN). A domain expires 35,999 blocks (200-250 days) after it is registered as a key-value pair in the Namecoin blockchain. Besides the domain registration (i.e., /d), Namecoin has a public online identity namespace (i.e., /id), along with other proposed services.

Despite its simplicity, the need to store the full blockchain to locate .bit domain addresses in real-time has discouraged Namecoin adoption. Although online explorer sites and browser extensions have been created to help Internet users with .bit domains, Namecoin namespaces have historically remained underutilized, and most blocks were empty of any new key-value pair [215].

A notable development in decentralized naming systems is the Ethereum Name Service (ENS), which provides human-readable domain names ending in `.eth` that map to Ethereum addresses and other on-chain resources. Unlike Namecoin, ENS is implemented entirely through smart contracts on the Ethereum blockchain, allowing seamless integration with wallets, dapps, and decentralized identity protocols. ENS names can be used to receive payments, host decentralized websites, or associate public metadata with an address. It also supports reverse resolution, enabling Ethereum addresses to resolve back to ENS names. Since its launch in 2017, ENS has become widely adopted across the Ethereum ecosystem, with millions of registered names and

growing compatibility with traditional DNS through support for domains like `.xyz`. ENS illustrates how decentralized naming can thrive when built atop programmable, widely used blockchain platforms.

Chapter Summary

- Proof-of-Work is the original consensus mechanism used in Bitcoin, but it has faced criticism due to high energy consumption and limited scalability.
- Alternative consensus mechanisms, collectively referred to as Proof-of-X, have been developed to address energy inefficiency. These include Proof-of-Stake, Proof-of-Burn, Proof-of-Authority, and Delegated Proof-of-Stake, each with distinct trade-offs in security and decentralization.
- DAG-based architectures, like IOTA's Tangle, offer scalable Layer-One alternatives to blockchains by validating transactions individually and in parallel.
- Sidechains allow asset transfers between a main chain and a secondary chain to improve performance and enable experimentation with different consensus models.
- Payment channels, such as the Lightning Network, are Layer-Two solutions that enable off-chain transactions between users, reducing latency and transaction fees.
- Rollups bundle off-chain transactions and post compressed data on-chain. Optimistic Rollups use a dispute window and fraud proofs, while zk-Rollups rely on cryptographic validity proofs for instant finality.
- Interchain technologies like Polkadot and Cosmos connect multiple blockchains via shared protocols, supporting asset and data interoperability through parachains, zones, and message-passing frameworks.
- Blockchain bridges enable cross-chain interactions but vary in trust assumptions. Trusted bridges depend on intermediaries, while trustless bridges rely on cryptographic mechanisms and smart contracts.
- Namecoin is a historical example of a blockchain-based domain name system. Despite its early promise, adoption has been limited due to practical constraints in usability and infrastructure.

Exercises

7.1 **Proof-of-Work consensus requires extensive computation. What is the main security rationale behind this design choice?**

(a) It guarantees faster block propagation across the network.
(b) It makes it prohibitively expensive for attackers to modify the blockchain history.
(c) It improves energy efficiency by limiting transaction throughput.
(d) It ensures equal mining opportunities for all nodes regardless of resources.

7.2 **Which scenario best illustrates the *nothing at stake* problem in Proof-of-Stake blockchains?**

(a) A validator supports multiple competing forks because doing so carries no financial risk.
(b) A validator loses their entire stake due to downtime.
(c) A miner wastes energy on an invalid block.
(d) A user's stake becomes illiquid during governance voting.

7.3 Is it possible for a blockchain to be both private and permissionless?

7.4 Give an example of a public and permissioned blockchain.

7.5 **In zk-Rollups, what ensures that off-chain transaction execution is trustworthy when posted to Layer 1?**

(a) A cryptographic proof that demonstrates correctness of state transitions.
(b) A challenge-response window allowing validators to contest the batch.
(c) A decentralized oracle service reporting transaction hashes.
(d) A trusted sequencer signing the final Merkle root.

7.6 **Why do zk-Rollups generally offer faster finality than Optimistic Rollups?**

(a) Because they skip all state validation on Layer 1.
(b) Because their fraud detection relies on community voting.
(c) Because they are only used for read-only smart contracts.
(d) Because they submit validity proofs that preempt the need for challenge periods.

7.7 **Which of the following are true about Delegated Proof-of-Stake (DPoS) systems? Select all that apply.**

- Token holders elect a limited number of delegates to propose and validate blocks.
- Voting power is often proportional to the amount of staked tokens.
- All users participate equally in transaction validation regardless of stake.
- DPoS systems reduce the likelihood of centralization.

7.8 Give two Proof-of-X alternatives to Proof-of-Work and explain how they work.

7.9 **Which of the following risks arises specifically due to the structure of DAG-based consensus systems like IOTA?**

(a) Sequential block confirmation leads to finality delays.
(b) Absence of a total ordering of transactions complicates double-spending detection.
(c) Miners control both consensus and fee distribution.
(d) Orphan blocks reduce throughput during high load.

7.10 **What is the role of a sequencer in rollup-based Layer 2 systems?**

(a) To periodically checkpoint blocks across Layer 1 and Layer 2.
(b) To verify fraud-proof submissions by rollup participants.
(c) To manage cross-chain token bridges between rollup networks.
(d) To determine transaction ordering and package them into rollup batches.

7.11 **In Cosmos, how is interoperability achieved between independent blockchains (zones)?**

(a) Through the Inter-Blockchain Communication protocol.
(b) By using cryptographic relays powered by zk-SNARKs.
(c) Through a relay chain, like in Polkadot.
(d) By atomic swaps over smart contracts.

7.12 **Which feature distinguishes Polkadot's architecture from most traditional blockchains?**

(a) Transactions are only stored off-chain.
(b) Block rewards are randomly assigned to parachains.
(c) Specialized blockchains (parachains) connect to a central relay chain for consensus.
(d) Each node in Polkadot runs its own consensus algorithm.

7.13 **Which of the following best explains why notary-based bridges are considered risky in blockchain systems?**

(a) They rely on zero-knowledge proofs, which are computationally expensive.
(b) They depend on a centralized set of actors to confirm cross-chain transactions.
(c) They require synchronization with DNS records.
(d) They can only be used between chains that share identical consensus algorithms.

7.14 **In Lightning Network, how does a Time-Locked Refund (HTLC timeout) protect users in the event of an unresponsive counterparty?**

(a) It permanently slashes the malicious user's balance.
(b) It locks the payment until the routing node acknowledges it.
(c) It routes the payment through an alternative path.
(d) It allows the sender to reclaim funds if the payment is not redeemed within a fixed time.

7.15 **Which of the following correctly describes the function of a two-way peg in blockchain interoperability?**

(a) It enforces slashing conditions across two networks.
(b) It synchronizes validator elections on both chains.
(c) It enables asset transfer between chains while locking equivalent value on the origin chain.
(d) It allows bridges to issue synthetic assets during high congestion.

7.16 **Which of the following best describes a key trade-off introduced by off-chain payment channels such as the Lightning Network?**

(a) They improve transaction throughput but reduce reliance on global consensus mechanisms.
(b) They guarantee on-chain finality for every transaction.
(c) They increase decentralization by eliminating the need for routing hubs.
(d) They enhance liveness by allowing transactions to be processed even when participants are offline.

7.17 **In bridge systems that use light clients or block headers for validation, why does storing only headers introduce trust assumptions?**

(a) Because light clients cannot generate zero-knowledge proofs.
(b) Because headers cannot be relayed between heterogeneous chains.
(c) Because bridge contracts cannot parse block headers directly.
(d) Because the header may commit to invalid state data that cannot be independently verified without the full block.

8

Decentralized Finance

his chapter introduces Decentralized Finance (DeFi), a rapidly growing area that enables financial transactions such as lending, borrowing, and trading to occur without traditional intermediaries. You will learn the foundational components of DeFi by exploring token standards, such as ERC20, ERC721, ERC1155 and ERC4626, stablecoins and their mechanisms for maintaining price stability, and wrapped tokens that facilitate cross-chain interoperability. The chapter explains how DeFi protocols are structured and how they function, focusing on core areas including decentralized exchanges, lending protocols, derivative markets, and yield farming. You will gain an understanding of key financial concepts adapted to blockchain systems, such as leverage, hedging, and impermanent loss. Particular attention is given to algorithmic innovations like automated market makers, flash loans, and synthetic asset creation. You will also study how oracles feed real-world data into smart contracts and how decentralized governance structures such as DAOs shape protocol evolution. Throughout the chapter, we emphasize the risks, incentives, and architectural trade-offs involved in designing and interacting with DeFi systems.

Decentralized Finance (DeFi) has emerged as a recent (and perhaps unforeseen) innovation that enables financial services such as lending, borrowing, and trading to be conducted on blockchains without the need for traditional financial intermediaries like banks and investment companies. Table 8.1 outlines the users and their roles in the DeFi ecosystem.

Werner et al. [401] describe the four defining characteristics of DeFi within the blockchain ecosystem as follows:

(i) Non-custodial: Participants have full control over their funds at all times, contrasting sharply with traditional finance, where institutions often hold and manage consumer funds.

(ii) Permissionless: Anyone, regardless of location or background, can access