Canopy Network

The Blockchain Incubator Model: Bridging Dependent Applications to Autonomous Networks

Adam Liposky / Andrew Nguyen / Shawn Regan
adam / andrew / shawn @canopynetwork.org

Draft 1.2 (December 2024)
Full Specification: github.com/canopy-network/canopy

Abstract. Fifteen years after the genesis of blockchain technology, the dominance of Layer 2 (L2) solutions for new blockchain applications highlights a paradox in scalability efforts: while many believe a multi-chain future is essential for technical maturation, most new projects are focused on retrofitting established ecosystems like Ethereum. This trend underscores the challenges of developing blockchain ecosystems that achieve full decentralization and autonomy. The absence of a framework that supports the full lifecycle, from entry-level dApp to fully autonomous Layer 1 (L1), remains a significant barrier to widespread adoption to blockchain technology. Canopy Network aims to solve this fundamental problem by providing a framework that offers a shared security service to blockchain applications with a seamless track to L1 independence. Application developers can utilize Canopy's unique consensus algorithm and architecture to become 'sub-chains', enabling access to critical validator infrastructure and unprecedented security mechanisms. By leveraging novel restaking economics, the model allows sub-chains to have immediate economic security, making adoption more accessible by replacing the need for substantial financial resources with community popularity. The plugin based architecture of the model allows an unprecedented one-way integration with external chains, allowing sub-chains to interact with networks outside the ecosystem without requiring any changes to the external chain software. This design addresses the limitations of existing projects by mitigating the constraints of ecosystem lock-in as well as the challenges and security risks of launching a fully independent blockchain. By providing a framework that emphasizes lifecycle evolution, technical simplicity, and financial inclusivity, Canopy Network introduces the first progressive autonomy solution for blockchain ecosystems.

Table of Contents

1. History

Bitcoin: The new paradigm

In 2009, the creation of Bitcoin [1] introduced a paradigm shift in the philosophy of technology, economics, and governance. Implemented as a peer-to-peer fully digital currency, Satoshi Nakamoto proposed a probabilistic solution to the 'Byzantine General's Problem' called Proof of Work (PoW) that guaranteed the integrity of distributed data without depending on one central authority. By using cryptographic principles and the physical limitations of brute-force hashing, the miners of Bitcoin were able to validate transactions and append blocks to the blockchain in a secure and trustless manner. In doing so, Bitcoin was able to coordinate a world-wide digital payment system among permissionless participants. This idea of trustless infrastructure by way of decentralized technology began the proliferation of cryptocurrency and the blockchain industry as a whole. While Bitcoin's radical architecture and decentralization solidified it as the father of blockchain technology - its design has many fundamental limitations. By only optimizing for transfer and store of value, Bitcoin stopped short of enabling applications beyond currency. A lack of programmability, energy efficiency, and scalability highlighted the need for innovation beyond Bitcoin's foundational framework.

Ethereum: Programmable money

Four years after the invention of Bitcoin, Vitalik Buterin expanded the vision of Nakamoto by publishing a paper titled *"Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform"* [2]. By evolving the blockchain state-machine from UTXO ledgers (Bitcoin) to multi-functional digital accounts, Ethereum enabled the creation of smart contracts, transforming blockchains from digital currency to dynamic ecosystems for generalized applications. This innovation introduced the idea of blockchain applications (dApps), allowing developers and enterprises to build automatic and decentralized services that leverage the advantages of blockchain technology such as censorship resistance and trustless coordination of permissionless actors.

Ethereum's innovation significantly increased its popularity, leading to an influx of transactions, resulting in congestion, high fees, and in some cases denial of service. While Gas fees (financial penalties proportional to the computational weight of each transaction) were meant to serve as a mechanism to manage resource consumption, the marketplace design induced volatile and unpredictable costs which reduced its usability and introduced significant financial barriers. This phenomena began years of research and development to 'scale' Ethereum - underscoring the shortcomings of a unichain design in addressing the scalability and accessibility needs of blockchain applications.

Tendermint: A modular BFT blockchain

Shortly after Ethereum, Jae Kwon introduced Tendermint: “a byzantine fault tolerant state machine replicator for arbitrary deterministic, finite state machines” [3]. Similar to Ethereum, he sought to make blockchain technology accessible to the masses for developing and deploying generalized blockchain applications. However, Kwon’s vision differed from Buterin by emphasizing a multichain model as the cornerstone of blockchain's future scalability and interoperability. To achieve this vision, Tendermint is not a standalone blockchain but rather an open-source, modular, consensus and peer-to-peer engine designed to streamline the development of L1 blockchains, by providing a foundational SDK for developers and enterprises to build upon.

In addition to its modular approach, Tendermint distinguished itself from Bitcoin and Ethereum by offering an energy efficient Consensus mechanism by adapting an existing solution to the Byzantine Generals problem, commonly known as Practical Byzantine Fault Tolerance. This, along with innovations by Peercoin [4] in 2012, resulted in the sybil protection mechanism known as Proof of Stake (PoS). With PoS, miners do not perform computational intensive brute force hashing - rather, they participate in stake-weighted voting, ultimately leading to them being reclassified as “validators”. Ethereum later adopted this concept, transitioning its network to PoS as a testament to its potential.

The creation of Tendermint marked a pivotal moment in blockchain history, as it made dApp independence feasible. However, despite its potential, Tendermint did not achieve the level of popularity seen with Ethereum. Unlike Ethereum - Tendermint's design did not address ease of deployment and secure bootstrapping, which are essential for fostering widespread accessibility and mass adoption. Although Tendermint chains avoided the scalability constraints inherent to Ethereum smart contracts, without robust financial backing, an experienced development team, and skilled node operators, successful independent deployments of Tendermint became increasingly rare.

Cosmos: Introducing interoperability

In 2016, the Tendermint project evolved into “The Internet of Blockchains”, when the Cosmos Network [5] and Cosmos SDK were revealed. The Cosmos Network was designed to provide a generalized communication framework called Inter-Blockchain Communication (IBC) to enable cross-chain functionality like the exchange of digital assets. The core product of Cosmos is facilitating multi-chain communication and data sharing between API compatible sovereign networks through their dedicated L1. This idea pioneered multi-chain interoperability - promising to bridge the gap between fragmented ecosystems, while enabling scalability through a connected network of fully independent Layer 1 blockchains.

Though Cosmos and the IBC protocol were groundbreaking advancements, they faced several challenges that prevented them from being a comprehensive solution for developer mass appeal. The primary concern with the Cosmos design is that, much like Tendermint, it is simply a development framework to launch L1s, enhanced with a standard communication protocol. While this approach certainly bolsters scalability, it suffers from many of the problems of Tendermint, not addressing bootstrapping challenges like lack of initial decentralization and economic security, and needing experienced network operators. Furthermore, while Tendermint and Cosmos offer a modular template for consensus and peer-to-peer, users of the Cosmos SDK must build their own state-machine and governance mechanisms, as well as actively maintain the consensus and peer-to-peer modules they did not originally author. As a result, while Cosmos provided the foundational tools for building interoperable blockchains, it still lacked the bootstrapping framework necessary for widespread adoption and ease of use in the blockchain ecosystem.

Polkadot: Shared security as a service

In 2020, Polkadot [6], created by Ethereum co-founder Gavin Wood presented a novel approach to blockchain scalability. Building upon lessons learned from Ethereum and Tendermint, Polkadot aspired to tackle the limitations of the existing systems by offering an interoperable, multi-chain, shared security model. At its core, Polkadot maintains a PoS base-chain that connects to various sub-chains, enabling them to operate separately while using security from the validator set of the base-chain. Similar to Ethereum, the sub-chain's state machine is uploaded to the base-chain so validators may be able to generically authenticate the sub-chain blocks, but like Tendermint, the sub-chain maintains its own transaction and block database - lessening unichain transaction bloat issues. In addition to enhanced scalability, Polkadot's multi-chain system enables cross-chain, interoperable communication, and shared governance between sub-chains, cementing a uniquely intertwined ecosystem.

While the idea of Polkadot is innovative, the design has many flaws that have greatly impacted its popularity and long-term success. Foremost, Polkadot attempts to achieve scalability through a paid service, meaning sub-chain access to Polkadot's security is limited and 'Slots' are auctioned periodically. This requires consumers to bond significant amounts of DOT tokens for extended periods, often causing liquidity constraints and creating high economic barriers to entry for sub-chains.

While Polkadot promises modularity of sub-chain operations, sub-chains are heavily dependent on the base-chain in a manner that is difficult to reverse. For instance, the design requires validators to execute sub-chain blocks using a Web Assembly State Machine (WASM) runtime binary on the base-chain. This forces each sub-chain to compile and upload its state machine to the base-chain (like a smart contract), necessitating programming language compatibility as well as careful coordination between the base-chain and sub-chains during any state-machine upgrades. Additionally, sub-chains rely on Polkadot's GRANDPA consensus mechanism for finality, meaning if sub-chain independence is achieved (which is not currently supported) *- the finality of historical blocks secured by Polkadot would be threatened and potentially reversible*. This means like Ethereum, sub-chains are lifetime dependent on Polkadot and reversing this reliance would involve significant architectural changes or a complete deprecation of the chain's history.

Furthermore, many challenges lie in using the complex architecture of Polkadot. Navigating the multi-tiered relationships between the base-chain, the sub-chain, interoperable communication, and complex governance functionality often demands expertise. For instance, sub-chains are also required to elect dedicated participants who are responsible for aggregating transactions and producing blocks, while other base-chain players ensure their validation. The prerequisite of understanding and burden of management creates a level of complexity that can be overwhelming for new users and developers, posing a significant educational barrier to entry. Moreover, complex built-in mechanisms for interoperability and systemic reliance on Polkadot governance underscore design choices that result in ecosystem-lock for its users. This presents challenges in achieving complete decentralization and autonomy for sub-chains as they mature, obstructing any transition to a self-sustaining, fully independent L1.

Rollups: The promise of scaling Ethereum

Around the launch of Polkadot, Rollups were popularized as the next-generation innovation for Ethereum scalability. In many ways, Rollups are similar to Polkadot sub-chains, offering a model that offloads heavy computation and state storage from the base-chain while retaining the security guarantees of the underlying protocol. Rollups achieve this by electing dedicated, often centralized, Sequencers who are responsible for aggregating transactions and producing blocks. However, Rollups are even more tightly coupled to the base-chain, than Polkadot sub-chains, as they post critical transaction data to ensure 'data availability' and maintain trust, especially given that Sequencers are typically centralized. The popularity of Rollups inspired projects like Celestia [7], led by Mustafa Al-Bassam, to champion data availability as an independent service, offering a Tendermint-based alternative for Rollup deployment. Unlike Ethereum Rollups or Polkadot's architecture, Celestia innovated by offloading transaction execution entirely to the sub-chains - further decentralizing computational responsibilities.

While Rollups promise scalability, they heavily detract from decentralization and introduce a complicated architecture. This type of implementation does not address ease of use nor graduation into fully independent systems, perpetuating concerns about scalability and long-term decentralization. Offloading block data to external protocols creates dependency and inefficiency, tying sub-chains to base-chains without a clear path to independence. Furthermore, data availability layers can exacerbate dependency concerns by requiring sub-chains to act as lite nodes to their own databases caused by the intermingling of all transactions on a shared layer. This design introduces inefficiencies and potential scalability issues for sub-chains as they require cryptographic proofs from the DAL to utilize their own data in a trustless fashion.

Avalanche: A marketplace of experienced validators

In 2020, Ava Labs introduced the Avalanche [8] protocol as a highly performant, multi-tiered blockchain model, that enables the launch of customizable independent sub-chains. Avalanche positioned itself as a scalability leader by offering a Directed Acyclic Graph (DAG)-like consensus algorithm that enabled high throughput and low latency, allowing thousands of transactions per second. Unlike Polkadot, Avalanche does not attempt to provide shared security, rather it serves as a platform for independent, API-compatible sub-chains to connect with a marketplace of credible (surety bonded) validators to run their sovereign blockchain. To access the marketplace, the sub-chain developer typically stakes AVAX on the base-chain - promoting itself as Avalanche compatible. This design prioritizes sub-chains to maintain their own security and sovereignty, employing the sub-chains to attract base-chain validators through their native token reward. While validators are always required to be staked on the base-chain, in most cases, 'hired' validators must stake on both the sub-chain and the base-chain layers, providing independent staking collateral for bad behavior on both the sub and base-chain. Taking further advantage of this model, Avalanche offers a native exchange platform, which enables token swaps between different sub-chains, creating a true competitive advantage to other less integrated ecosystems.

While there are many promising features of this protocol, some inherent challenges have contributed to a relatively lower adoption compared to competitors like Polkadot. While the fundamental design of Avalanche focuses on sub-chain sovereignty, there are questions regarding the value proposition. The core architecture of Avalanche relies on sub-chains attracting and rewarding base-chain validators with the sub-chain's native asset. This raises questions about why sub-chains would use Avalanche. *Is any security improved from this design if the 'hired' validators may switch allegiances for monetary value greater than the sub-chain's token reward*? Furthermore, base-chain stake cannot be slashed by the sub-chains (in fact stake cannot be slashed at all in the base-chain) - blurring the lines between the appearance of safety and actually providing security.

Though API compatibility does lower the barrier for validators to support a sub-chain, Avalanche validators are typically required to stake on both the base-chain and the sub-chain. This raises concerns about how much Avalanche truly simplifies validator adoption for the end user, highlighting a lack of mechanisms to address financial barriers in the early stages of the blockchain lifecycle. The issue is further compounded by requiring users to buy the native AVAX token and stake it to show 'skin in the game' to promote itself within the Avalanche ecosystem. Additionally, since the base-chain does not implement consensus or peer-to-peer for the sub-chain, it leaves developers responsible for creating their own consensus and networking layers. This implementation requirement is a high technical barrier to launch in comparison with other models.

Though Avalanche provides a template for sub-chains, by default it only offers the DAG-like consensus algorithm, which is more complex and experimental than widely adopted alternatives like BFT. This underscores challenges for ease of implementation and management. Though the protocol does have the benefit of a native token exchange within the Avalanche ecosystem, it does not have much interoperable functionality outside of Avalanche compatible sub-chains, presenting challenges in long term sustainability and mass adoption.

EigenLayer: Restaking economics

Launched in 2023, a series of smart contracts on Ethereum called EigenLayer [9] introduced a novel economic model called 'restaking'. Restaking allows Ethereum validators to reuse their bonded tokens to secure additional services and applications beyond just Ethereum's base layer. By leveraging Ethereum's consensus mechanism and Validator set, EigenLayer enables decentralized projects to access an Ethereum based (opt-in) shared security model. Although this concept is similar to that of Polkadot and Rollups, the restaking economic model is innovative as it enables validators to secure sub-protocols using their already-bonded tokens by permitting the re-use of collateral. By doing this, a more capital-efficient architecture is produced, allowing validators to use their current stake to offer the service while lowering the cost for blockchain apps to access a shared security model. However, while EigenLayer's approach is useful for Ethereum DApps, it is not applicable to other systems because its restaking economics are exclusive to the Ethereum ecosystem. Thus, EigenLayer only addresses scalability through L2 shared security services that utilize restaked Ethereum collateral. Furthermore, EigenLayer does not attempt to address interoperability, leaving these challenges to be solved by other technologies.

Blockchain Applications: The basics

A blockchain application holds several critical advantages to traditional web apps by providing enhanced features that redefine how users interact with digital platforms:

- **Decentralization:** No single entity controls the data or operations, limiting single points of failure and preventing censorship.
- **Permissionless and Trustless Interactions:** App interactions rely on cryptographic proofs and immutable rules, eliminating the need for trust in other users or counterparties.
- **Transparency:** All transactions are recorded on a public ledger, allowing for auditable and verifiable operations.
- **Ownership and Control:** Blockchains are democratically driven systems, enabling governance and fair distribution of power.
- **Immutability:** Application data written to a blockchain cannot be altered, enhancing availability and accountability.

Ultimately, these properties are achieved by hosting the application on a network of peers (validators) that use a consensus mechanism to agree on the state of the application. The validators are able to interact trustlessly due to a predefined protocol that leverages cryptography for verifiability and integrity.

Layers of Blockchain: L1 and Dependent Applications

The definitions being offered in this paragraph are as commonly understood by the industry today. Within this paper, DApps, L2s, Parachains, Side-Chains are all categorized as "Dependent Apps" to highlight their shared reliance on a base-chain for critical functions such as security, consensus, and governance.

Blockchain Layer Definitions

- **Layer 1 (L1):** An independent protocol that operates on a network of peers (validators), who use a consensus mechanism to reach agreement on the state of an application. Examples include Ethereum and Polkadot.
- **Dependent Applications:** Applications or protocols that rely on an L1 for execution, security, or scalability
  - **Layer 2 (L2):** An application or protocol built on an L1 to enhance the base application in scalability, functionality, and/or speed. Examples include EigenLayer and Rollups.
  - **Decentralized Applications (DApps):** Applications that upload their execution code to an L1 or L2 protocol to leverage the underlying protocol to achieve decentralization.

L1: The cost of independence

One of the reasons Dependent Apps exist is that L1 blockchains are difficult to create. Foremost, high economic value and a diverse set of validators are prerequisites to blockchain application security. Due to the permissionless properties of a blockchain application, a low value, weakly decentralized blockchain applications are susceptible to hostile takeover. This is commonly known as a 51% attack in bitcoin or a 66% attack in BFT based networks.

Because of this vulnerability, most projects that achieved success with an L1 have been required to have enormous fundraising events in order to personally guarantee the security of their blockchain application. Some notable examples are listed below [10]:

Fundraising Prior to Launch

- **Polkadot:** $144,000,000
- **Avalanche:** $60,000,000
- **Solana:** $25,000,000
- **Ethereum:** $18,300,000
- **Cosmos:** $17,000,000

Additionally, a L1 system is generally considered difficult to develop. A Layer 1 blockchain typically consists of a (1) predefined application (state machine) whose state is determined by validators via a (2) consensus mechanism, communicated through a (3) peer-to-peer network, and stored in a verifiable manner, often using a structure like a (4) Merkle tree. Building such a system in an emerging industry can be considered challenging for developers who are unfamiliar with the complexities and nuances. [11] [12] [13] [14] [15]

Development Time to Launch

- **Polkadot:** 4 years
- **Avalanche:** 2 years
- **Solana:** 3 years
- **Ethereum:** 18 months (+ 6 years for Eth2)
- **Cosmos:** 3 years

Unichain Model: The cost of dependence

As described in the whitepapers of projects like Ethereum and Polkadot, their objective is to allow developers to build blockchain applications easily. The idea is that building on an existing L1 network can abstract away the economic and technological barriers of starting a blockchain application by enabling the use of the decentralized properties of the existing network. The culmination of this idea is seen with Dependent Apps, which ultimately allows decentralization by proxy. This solution enables users to have immediate economic security by having the validators of the underlying L1 come to consensus on the state of the Dependent Application as they produce the next block. Additionally, the developer only needs to focus on creating the application, not worrying about the decentralization of the application through a peer-to-peer and consensus layer, as these are already handled by the L1 blockchain.

However, when considering the blockchain application ideal, Dependent Apps are inferior to Layer 1s in many ways. While Dependent Apps are perceived to be convenient alternatives to L1s, their classification as decentralized technology warrants closer examination. Given that developers of blockchain applications are inherently driven by the principles of decentralization, it is seemingly important to challenge the thesis of Dependent Apps and how their reliance on underlying L1s for security, governance, and consensus inherently centralizes critical functions.

	Layer 1	**Dependent Apps**
Decentralization	L1 blockchain networks are completely autonomous	Dependent Apps introduces a single point of failure with an inherent dependency on an external protocol
Permissionless & Trustless Interaction	L1 users (validators) directly provide and interact with the consensus mechanism, enabling trustless interactions	Dependent Apps users rely on a third party to enable permissionless interaction and often introduce semi-centralized components for core functionality
Transparency	L1 blockchains directly record their transactions on their fully owned blockchain, ensuring full verifiability and data availability	Dependent Apps relies on an external service to maintain and provide the data
Ownership & Control	L1 protocols are completely sovereign entities, enabling democratic control directly by the users of the application	Dependent Apps typically face major barriers to upgrading their application and have limited representation in governance
Immutability & Data	L1 blockchains directly record their transactions on their fully owned blockchain, ensuring immutability	The app can’t independently manage its historical data, leading to permanent dependence on the host. This occurs because block links are disrupted by unrelated data

Most of the advantages blockchain applications have over traditional apps is the features produced by decentralization. It is fundamentally true that any system that depends on a central authority or host protocol to operate does not hold the properties of decentralization. Though Dependent Apps are often promoted as decentralized, their reliance on L1 networks may compromise their qualification as a decentralized technology. True decentralization requires the ability to govern, secure, and scale independently, without relying on external services. This dependency exposes Dependent Apps to vulnerabilities related to the governance and consensus mechanisms of the L1 network. For example, when hosted on a platform like Ethereum, Dependent Apps are subject to the decisions and policies of Ethereum's validators, which can centralize control and influence key operations, such as transaction finality and the risk of censorship. While it has been taboo to challenge the decentralization of DApps/L2s in the past, recent events underscore the reality of the relationship between Dependent Apps and the underlying protocols they rely on.

As an example of censorship, in 2022 [16], the Ethereum community faced pressure from regulators, including the United States government, to comply with sanctions imposed by the Office of Foreign Assets Control (OFAC). This pressure resulted in a situation where Ethereum validators began censoring transactions from a specific DApp, Tornado Cash, in order to adhere to OFAC standards. At its peak, more than 50% of Ethereum blocks included only OFAC-compliant transactions, demonstrating a coordinated effort among validators to censor transactions that did not meet these regulatory requirements. Vitalik Buterin, founder of Ethereum, advocated for this behavior - calling out to the public to "tolerate the censorship", sparking controversy. Many community members from Tornado Cash's ecosystem publicly disagreed with the censorship - raising concerns about the ethical and technical implications. Since Dependent Apps and L2s depend on a host protocol, they are ultimately disenfranchised as non-autonomous entities and are able to be censored by a central group of non-stakeholders.

In addition to the notable detractors of the core offering, there are several critical scalability challenges of a unichain model like Ethereum. Currently, all unichains are designed to support their Dependent Apps indefinitely, leading to a single bottleneck for many dependent systems. If successful, Dependent Apps are unable to scale on a unichain like Ethereum due to the shared nature of the network's resources. For example, all Dependent Apps compete for the same computational power, storage, and bandwidth and high demand from one application can cause network congestion. This ultimately causes increased transaction costs and slower processing times for other users - especially as Dependent Apps achieve success and grow.

This phenomena is highlighted by one of the most successful applications ever built on Ethereum: CryptoKitties. In 2017 [17] [18], CryptoKitties became infamous for contributing to network congestion. The high demand for the application led to high gas fees and slow transaction times on Ethereum, frustrating users and making it difficult to play the game or transact using other Dependent Apps on Ethereum. At a critical juncture, the Ethereum scalability limitations hindered the growth of CryptoKitties, ultimately leading to a crisis from success.

Another notable example was Decentraland [19], a DApp built on Ethereum, whose success was stunted by the scalability challenges of the unichain model. Just like CryptoKitties, the popularity of the application coupled with the shared resources design of Ethereum led to financial and technical bottlenecks - providing an unacceptable user experience at a pivotal moment for its growth.

CryptoKitties, among other congestion events, eventually led to changes in the roadmap for Ethereum. These examples are far from isolated; Ethereum's inability to support all of its applications sparked an entire industry of L2 technology to create retrofit solutions to the scalability issues. However, these Layer 2 solutions often come at the cost of increased complexity and a reliance on centralized operators, further complicating the ecosystem. For example, Rollup solutions require trust in the operators (collators / sequencers) for transaction finality, which is a critical detractor to trustlessness and censorship resistance. Additionally, many require off-chain infrastructure that can be more vulnerable to centralization or attacks. This further diminishes the decentralized ideals that blockchain applications aim to uphold.

Autonomy: A fundamental requirement

True decentralized technology operates autonomously - without any reliance on an external authority or protocol. By definition, this is only possible via creation of an independent L1 blockchain that implements its own consensus, peer-to-peer, and governance protocols. Such an implementation operates free from governance and censorship vulnerabilities as well as the issues of a shared resource environment, enabling scale without external constraints.

For many existing blockchain applications, unhindered success can only lie in implementing and transitioning to a fully self-sufficient L1. Autonomy at the L1 level provides both technical and foundational benefits. It ensures that the core advantages of a blockchain application are maintained, while also offering the ability to evolve to meet user demand. As the application grows, the network is able to scale without the restraint of dependency - enabling developers to maintain acceptable user experiences and service level agreements. However, this ideology is about more than just enhancing performance, instead it is about ensuring long-term sustainability, free from the issues and taxes imposed by centralized or semi-centralized systems. It ensures that control remains decentralized, as governance is in the hands of the network's users, enabling a trustless and permissionless environment. This drive for autonomy has led many successful Dependent Apps to seek an escape from the unichain they were built on. However, the cost of escaping a unichain is impossibly high.

Migration: The costs of escaping a unichain

Migrating away from a unichain is not just an inconvenient process; it requires a fundamental shift in the design of the application itself. There are critical, irreversible dependencies when building a layer above a unichain that causes severe pain-points if an application 'outgrows' the underlying L1. Due to their fundamental design, Dependent Apps are bound to the base protocol for critical components like historical finality, governance, and data availability. In fact, the dependence is typically so deep that historically it has triggered complete rewrites of the technology coupled with a socially coordinated migration and centralization compromises in the new stack to achieve the necessary scale for a successful application.

In 2017, CryptoKitties [17] [18] was dying from success. Ethereum was unable to keep up with their success and their reputation was under pressure for 'causing' cascading failures for other Ethereum applications. The CryptoKitties creators knew they had outgrown the Ethereum 'launching platform' but there was no path to scale beyond Ethereum. This ultimately led them to completely pivot, forcing them to develop their own L1 blockchain 'Flow'. The process of development and migration ultimately took three years, leading many to wonder if they missed their opportunity for greater levels of success.

In 2021 [19] Decentraland was causing heavy network congestion and high transaction fees for Ethereum's DApp community. Like CryptoKitties before them, Decentraland's leadership understood they had outgrown the Ethereum infrastructure. However, unlike CryptoKitties who faced the cost of building a new L1 from scratch and rewrote all of their tech, Decentraland announced they would pivot their technology to Polygon, (formerly Matic) an L2 solution, to achieve the scale they required and to alleviate resource usage for other Ethereum applications. Despite the strategy, this too caused significant technical rewrites to be compatible with Polygon, as all transactions had to be rerouted to the L2, even though Polygon was designed to be compatible with Ethereum applications. Additionally, using an L2 meant Decentraland had to compromise their decentralization further - relying on Polygon's centralized sequencer for essential operation. The gradual migration that started in 2021 is still ongoing three years later, the ending timeline for this is unclear.

It is important to note that these incidents are not isolated, as there are many more examples of former Dependent Apps that were forced to make similar transitions: [20][21][22][23][24][25]

Blockchain Migration Examples

- **Audius:** Ethereum → Solana
- **Axie Infinity:** Ethereum → Sovereign Side-Chain
- **Celo:** Ethereum → L1 → L2
- **Synthetix:** Ethereum → L2
- **Tether:** Ethereum → Tron + Multi-Chain
- **Uniswap:** Ethereum → L2 / Multi-Chain

Building a blockchain in 2024: The dilemma

There are two typical paths to build a blockchain application:

Dependent Apps

– **Pros:**

- Low-cost, immediate security integration
- Access to established ecosystem and infrastructure
- Focus on building the application layer without managing blockchain-level issues

– **Cons:**

- Dependence on L1 for security, finality, and consensus
- Compromised decentralization, as operations may be controlled by few validators
- Shared resources causing potential congestion and competition
- Limited governance capabilities and upgradability

L1

– **Pros:**

- Full sovereignty and self-governance
- Native security and independent scalability
- Long-term potential for ecosystem growth and development

– **Cons:**

- High cost and complexity to launch and maintain
- Requires large infrastructure and validator network
- No immediate ecosystem or user base

Currently there is no solution that bridges the gap between these two paths. *For developers, this is a true dilemma.*

To build an L1 from scratch a huge fundraising effort must take place to ensure financial security of the protocol from hostile takeover attacks. Additionally, a heavy development and research phase must take place in order to ensure technical safety and product quality. Furthermore, an experienced set of validator operators must be acquired to ensure true decentralization and maintain operational stability. A brand new ecosystem must be fostered and user based tooling must be developed.

To build a DApp compromises any core 'blockchain' offering as it is inherently dependent and controlled by the host protocol. Moreover, to build a DApp is to accept the success of the application is capped from the beginning. A unichain model guarantees a ceiling to throughput and scale as the resource usage is fundamentally shared between all Dependent Apps on the unichain. Not to mention the reputational risk assumed when an application achieves success, detracting from success of other applications without a quick solution to alleviate the pressure. On top of that, the shared resource environment is life-long and irreversible and the path to scale often involves heavy technical rewrites, socially coordinated ecosystem shifts, and further compromises in the core offering with semi-centralized components. Moreover, the value captured from an DApp built on a unichain is primarily captured by the unichain in the form of gas fees, not the application creating the value. The only other alternative is starting to build an L1 from scratch, which comes with the pain-points described above.

Beyond Decentralization and Scalability

Beyond the immediate benefits of scalability, autonomy, censorship resistance, transparency, and control decentralization/L1 deployment offers tangible benefits to developers:

1. Customization - Developers are able to customize their state machine to their needs, rather than inheriting these specifications from a base-layer.
2. Economic control - Developers are able to introduce innovative value capture mechanics not available as a DApp.
3. Efficiency: Applications built on a dedicated L1 blockchain have exclusive access to their own block space and validator network, avoiding the resource competition inherent in unichains.
4. Upgradability - Developers are able to upgrade their application without coordination with a host protocol.
5. Flexibility of deployment - With L1s like Ethereum, developers are limited in the ways that they can build and must conform to the EVM standard. L1s provide the flexibility to build in the way that the developer chooses.

Vision: The ideal future

Projects like Ethereum and Polkadot recognize the importance of lowering barriers to blockchain application development for wider adoption. By offering a 'quick to market' solution that does not require extensive fundraising or excessive centralization, they helped the industry grow. However, these solutions are highly limited - providing success-capped alternatives whose core offerings are diminished by the very nature of their design.

The ideal vision for future blockchain application development would be a platform that allows applications to start with a dependent architecture to make bootstrapping accessible, but allows gradual scalability and progressive decentralization to a full L1 network to eliminate those dependencies once mature. This design would remove the financial and technical barriers to entry, while pre-defining a track to sovereignty to eliminate the significant costs of escaping the host protocol. There currently does not exist a solution that bridges the gap between Dependent Apps and autonomous L1 networks.

2. Technical Introduction

Preamble: The missing layer

An analysis of the advancement of blockchain technology in the past fifteen years demonstrates a narrow industry focus - rather than fostering diverse, independent ecosystems. While initially projects pursued user-centric adoption, the trend of the industry has shifted to unichain throughput. This phenomenon has left certain trade-offs unresolved: supporting the entire application lifecycle, from bootstrapping to full sovereignty, remains unaddressed; accessibility is hindered by complexity, and securing a blockchain requires substantial economic resources and operational expertise. The blockchain incubator model attempts to address these challenges, offering a unique model to simplify the journey from an entry-level dApp to a fully independent L1. By implementing a user-focused growth path that prioritizes lifecycle evolution, technical ease, and economic accessibility, Canopy Network provides the first progressive autonomy framework for blockchain ecosystems.

Introducing: Canopy Network

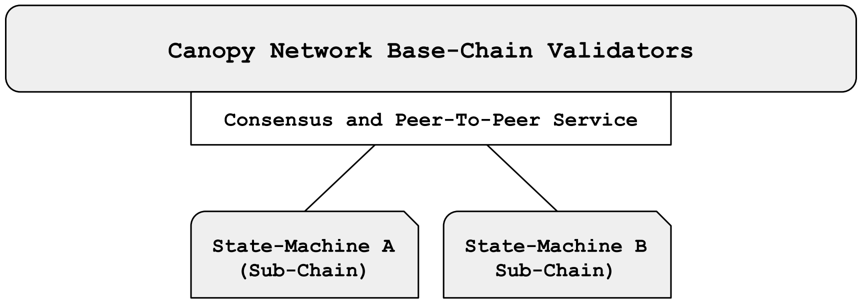
Canopy Network's incubator model is designed to launch and grow new blockchain ecosystems, enabling small applications (sub-chains), to grow into fully sovereign Layer 1 networks. Unlike existing solutions that often cause lifelong, irreversible dependencies or require significant technical or financial resources, Canopy offers an accessible, capital-efficient alternative.

To accomplish this, Canopy offers a shared security service that is powered by a novel restaking economic model along with a fully programmable, modular framework for blockchain development. Through a novel consensus algorithm and unique design, the model introduces advanced mechanisms such as 'chain-halt rescue' and 'long-range attack' mitigation - enhancing the security of its sub-chains. The model's flexible plugin architecture enables one-way integration with projects outside of its ecosystem, enabling unprecedented ease of interoperability. Additionally, Canopy employs innovative security mechanisms to ensure the reliability and stability of its sub-chains, allowing developers to build scalable, independent networks. The following sections provide a technical introduction to the protocol.

The protocol: An overview of key concepts

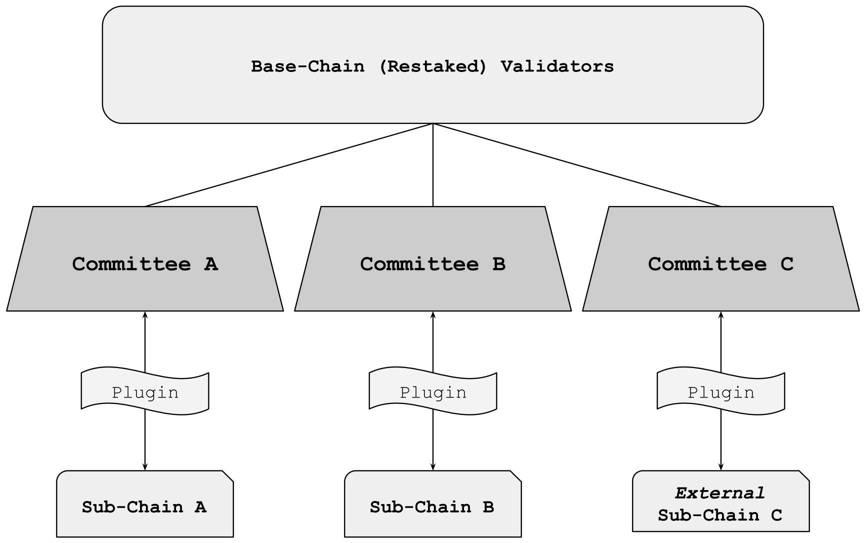
Canopy offers a unique combination of features that enable progressive blockchain ecosystem autonomy. This section offers a technical introduction of how it works:

Canopy provides blockchain developers consensus and peer-to-peer layers for sub-chains through a dynamic set of base-chain validators. Validators stake the protocol's native cryptocurrency (CNPY) and are able to reuse the same collateral to validate multiple chains, improving capital efficiency. Canopy uses a plugin-based communication model, where validators interact with privately hosted sub-chain nodes through bilateral APIs.

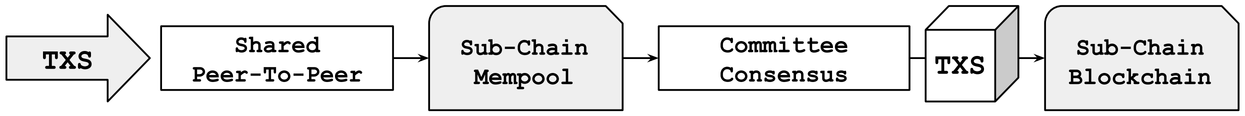

Canopy Network Base-Chain Validators
Consensus and Peer-To-Peer Service
State-Machine A (Sub-Chain)
State-Machine B Sub-Chain)

This design offloads state machine operation and block storage to the sub-chain software, while managing the consensus and peer-to-peer layers of the process at the base-chain layer. Crucially, this architecture facilitates eventual sub-chain graduation to independence with a predefined track.

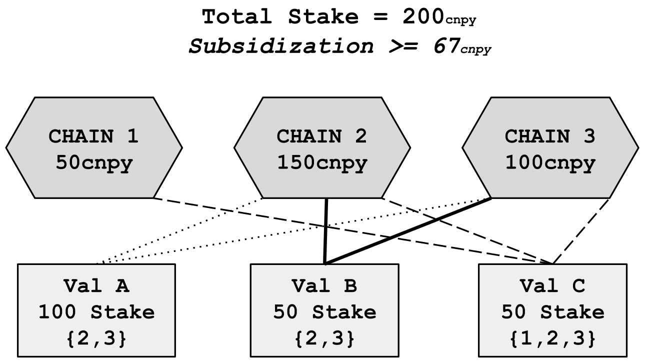
This unique plugin architecture extends beyond just new sub-chains, enabling permissionless integrations to existing chains without needing modifications to their software or protocol. Validators using the same plugin are organized into committees, who work together to achieve sub-chain consensus.



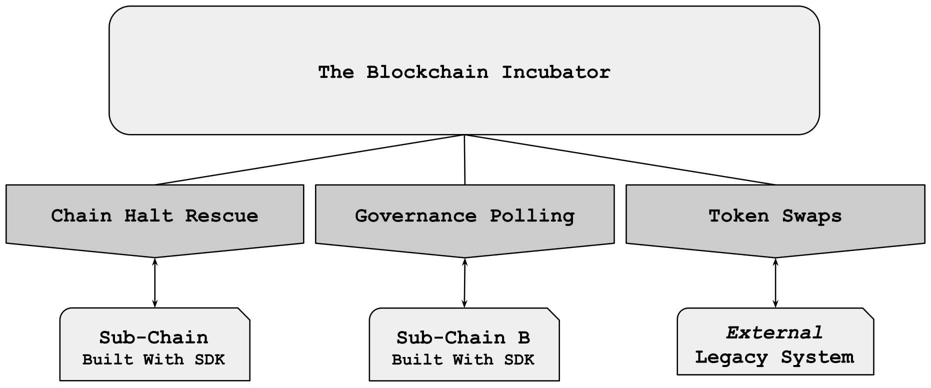
Unlike existing solutions, sub-chain block production and transaction aggregation does not require any specialized actor. Instead, sub-chain transactions flow through the shared network to their respective mempools, where a block producer aggregates them for consensus using NestBFT. After each sub-chain block is committed, the block producer submits a summary transaction to the base-chain, reporting information about what happened during the process, leading to things like reward distribution and stake slashes.



Base-chain rewards follow a fixed inflation schedule and are divided equally among subsidized committees. A committee's subsidized status is based on the amount of validator tokens committed to each on the base-chain. Meaning, sub-chains are able to have a dedicated committee if enough validators are willing to restake their collateral on its behalf, shifting requirements to fund blockchain security from financial investment from the project to community popularity.



Canopy also provides a pre-packaged state-machine software template for sub-chains, complete with tools like a web wallet, blockchain explorer, and governance tooling for low-barrier, rapid development. Furthermore, while the incubator's primary focus is to provide a platform for the lifecycle evolution of blockchain applications, it does not compromise on ecosystem interoperability, providing native token swaps and multi-tiered validator driven governance.



3. Sub-Chain Integration

Plugins: Modular integration

A plugin serves as a lightweight interface between the sub-chains and the validators of the base-chain - enabling flexible construction and sovereign upgradability. Each plugin bridges both software via a simple API:

Algorithm 1 Plugin Interface

- `propose_block`: get txs from the sub-chain mempool to propose a block
- `validate_block`: validate a peer block by interpreting the txs
- `store_transaction`: saves a transaction to the sub-chain's mempool
- `store_block`: saves a block to the sub-chain block store
- `load_block`: loads a block from the sub-chain block store, enabling peer sync through the shared peer-to-peer network
- `start_bft`: begins the committee BFT instance, allowing the sub-chain to control its block times

By executing communication with a bilateral API and providing sub-chains with an out-of-the-box template, the plugin enables sub-chains to access a shared security model with minimal complexity. This approach combines the modular flexibility of Tendermint, without the insecurity of bootstrapping, and the shared security model of Polkadot, while avoiding its high complexity and rigid framework.

Interoperability: External chains

Canopy's architecture is designed not only to integrate with new sub-chains but also plug into existing systems including legacy chains. This provides the ability to create generalized side-chain applications without requiring any modification to the existing chain software or protocol. Below is an example of a simple plugin that rewards validators for quality on an existing network:

Algorithm 2 QOS Validators for Tendermint Plugin (Example)

- The plugin notifies committee members of a new Tendermint block to start the consensus process.
- Each member reads the latest Tendermint block to identify which validators signed it.
- The committee reaches consensus on the list of signers.
- The committee then directs a portion of their CNPY block reward to the addresses that match the signers' public key on Tendermint.

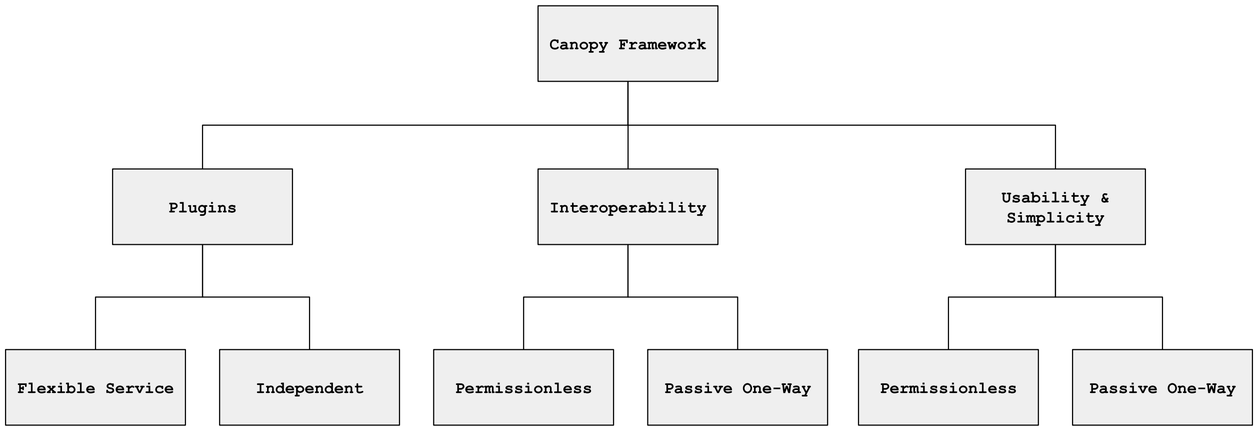
This example demonstrates how Canopy's plugin architecture seamlessly integrates with existing blockchain systems to enhance their functionality and incentivize network participation. To further illustrate the point, below is a list of some applications that may be built by creating a plugin for an existing system:

1. **Interoperability**
   - Cross-chain bridges
   - Native token swaps
   - Shared liquidity applications
   - Unified governance
2. **Scalability extensions**
   - Lightning network
   - Sharding network
   - Side-car dApp deployment
3. **Security extensions**
   - Chain-hault fallback
   - Dispute resolution framework
   - Checkpointing-as-a-service
4. **Governance extensions**
   - DAO coordination
   - Cross-chain voting
   - On-chain polling
5. **Decentralized oracle**
   - Chain state reporting
   - Off-chain data validation

Enhancements: Plugin driven innovation

Thus, the incubator's plugin model offers greater flexibility than systems like Ethereum's solidity framework, by providing a more modular - customizable approach to smart contract development. Unlike Ethereum, which permanently anchors a developers implementation to the ecosystem, Canopy's design allows for a dynamic and programmable state machine template that is easily adapted to meet the utility of various blockchain applications. This architecture allows developers to deploy a wider range of decentralized applications without being lifetime dependent on an ecosystem.

Similarly, Canopy's plugin framework offers greater usability and simplicity than Polkadot's Web-Assembly State Machine (WASM) architecture. By requiring sub-chains to compile and upload their state-machine to the base-chain - Polkadot's runtime design introduces deployment complexity and network coordination. In contrast, Canopy implements a straightforward plugin architecture for cross-chain communication and consensus, ensuring sovereign upgrades without base-chain coordination. This not only simplifies the development but also reduces overhead, making Canopy the ideal solution for teams that would rather focus on their application logic than a coordinated plan for upgradability.



Additionally, Canopy's interoperability model surpasses Cosmos in ease of integration. Instead of requiring major protocol upgrades to be IBC compatible, Canopy offers a passive, one-way integration model. This allows the external project to transition to - or be enhanced by the blockchain incubator model without any protocol upgrades. This flexibility offers a more realistic path to an interconnected 'Internet of Blockchains'.

4. Canopy Base-Chain

State-Machine: Accounts and validators

The foundation of Canopy's state-machine is an account based, validator staking protocol. The native crypto-currency (CNPY) is able to be transferred to and from accounts and surety bonded by participants to become validators. When staking (bonding), validators are able to choose a variety of configurations:

Stake Transaction

- **amount:** The amount of tokens to be removed from the sender account and locked as a surety bond against bad behavior.
- **committees:** The list of committees the validator is restaking their tokens towards.
- **is_delegate:** A non-delegate is registering for active participation in the committee consensus, whereas a delegate is passive participation that contributes its tokens to influence subsidized status.
- **is_compounding:** Automatically compounds rewards to the stake or early withdraws them to the reward address for a penalty.

The more tokens the validator bonds, the more voting power it has in consensus operations. However, the more staked, the more at risk if 'slashed' for bad behavior like double or non-signing during consensus.

Once registered, a validator is eligible to provide shared security services to any committee it chooses. In Canopy, the base chain functions as just another sub-chain - leveraging the network's shared security. This design allows validators to secure other sub-chains without being required to validate the base chain itself. Once in a committee, the validator connects with the other members of that committee through the shared peer-to-peer layer. Together the committee executes consensus for the sub-chain, producing immediately final sub-chain blocks. After each consensus round, the elected leader submits a summary transaction to the Canopy base-chain, encapsulating the results of the round.

Summary Transaction

- **reward_recipients:** Who the committee agreed to distribute the block reward to.
- **slash_recipients:** Who the committee agreed to slash due to evidence of bad behavior.
- **orders:** Details of processed swaps for cross-chain liquidity.
- **checkpoint:** A verifiable sub-chain snapshot for Checkpoint-as-a-Service.

At any time a validator is able to update their stake information like committees and compounding status by submitting an edit_stake transaction:

Edit Stake Transaction

- **amount:** The updated amount of tokens being staked. This must be greater than or equal to the previous staked amount.
- **committees:** The update to the committees the validator is restaking their tokens for.
- **is_compounding:** The update to the auto-compounding status.

In addition to these transactions, the validator may also pause operations across all committees by submitting a `pause_tx`, temporarily removing it from service, and submit an `unpause_tx` to resume committee membership. To exit completely the validator may submit an `unstake_tx`, permanently leaving the committees it's staked for. For short and long-range attack protection, the validator must wait for governance-controlled `unstaking_blocks` before their bond is fully returned, during which they are still eligible for slashing for bad behavior.

Token Swaps: Internal and external

Canopy is able to offer token-swaps to any internal or external sub-chain without requiring changes to the sub-chain software integration. This is possible due to committees being trustless oracles for both the base-chain and the sub-chain. To begin the swap process, an asset seller may transfer funds to an escrow account controlled by the committee. This transaction includes meta-data like an exchange rate for the sub-chain-asset and the seller's recipient address on the sub-chain. The committee observes any acceptance of the ask on the sub-chain through plugin specific signaling like memoization that announces the buyer's receive address. Once accepted, the committee locks the escrow funds and awaits for the buyer to transfer the sub-chain asset to the seller's address. Once witnessed, the committee releases the escrowed funds to the buyer's address. Below is a more detailed description of the swap protocol:

Canopy Token Swap Protocol

The Swap Protocol facilitates a token swap between Alice (with Token A) and Bob (with Token B). The committee oversees the process while controlling the escrow account of Blockchain A and observing Blockchain B.

1. Alice creates a *SellOrder* with the amount of *Token A* she wants to sell, the *exchange rate*, and her *Token B address*. Token A is escrowed in a committee-controlled address. Alice can reverse this order by submitting a transaction on Blockchain A.
2. Bob accepts Alice's offer by sending a transaction on Blockchain B, referencing Alice's offer hash and providing his *Token A address* in the memo field.
3. The committee updates the recipient of Alice's sell order to Bob's *Token A* address, verifying that Bob has enough *aged Token B* in his Blockchain B address.
4. Bob sends *Token B* to Alice, with a memo linking to the *Request to Sell*.
5. The committee witnesses Bob's transaction and releases Alice's *Token A* to Bob.
6. If Bob does not send *Token B* within N Blockchain B blocks, the committee resets Alice's *Request to Sell*.

It is important to note that memoization does not have to be in a 'memo-field' but can be signaled in any chain-specific way, like Bitcoin's `Signature Script` or Ethereum's `Data` field.

Chain Halt Rescue: A unique value proposition

Due to the decoupled nature of consensus management and sub-chain operation, the Canopy protocol is able to offer a unique value proposition: chain-halt recovery. Chain-halt typically occurs when there's a code error that leads to non-determinism in the state - preventing nodes from coming to consensus on the state of the ledger. Another common reason chain halts occur is a lack of a validator supermajority due to faulty validators. However, in Canopy, the base-chain manages validator committees for the sub-chains, ensuring that even if a sub-chain encounters a critical bug or validator failure, the base-chain can reassign validators, restart consensus, or enable state rollbacks. This design emphasizes the robustness and fault-tolerant design of Canopy's architecture.

5. Consensus Background

Weak Subjectivity: The challenges of Proof of Stake

In 2014, Vitalik Buterin published a blog post [26] titled: "Proof of Stake: How I Learned to Love Weak Subjectivity". In this paper, he describes how while PoS is an inherently more energy-efficient mechanism than its predecessor, it introduces the concept of a "Long-Range-Attack". An attacker executing a Long-Range-Attack uses old validator keys, that once controlled a super-majority of the validator set, to rewrite the chain's history from a point far in the past. This is possible because the economic value that the keys once held has diminished and the attacker is presumably able to acquire them cheaply. Using these keys, an attacker is able to quickly rewrite the forward history of the blockchain to become the 'longest chain' and potentially convince peers to join the fork enabling economic attacks. In his post, Vitalik proposed the use of "weak subjectivity", requiring node security to rely on social consensus by way of external checkpoints (finalized block hashes associated with specific heights) to mitigate this risk. He suggested checkpointing every 12.8 minutes for Ethereum, aligning with the network's epoch system. At the time of writing, this approach is widely regarded as one of the only effective mechanisms to prevent Long-Range-Attacks.

However, checkpointing as a mitigation for Long-Range-Attacks has significant shortcomings. First, the design of constantly publishing and downloading checkpoints to validate a safe state is an administrative burden that may lead to participants ignoring the requirement - thus diminishing its effectiveness. More critically, the network security of a system is anchored to community accepted checkpoint sources like software deployments and social media. This dependency heavily increases trust assumptions and makes the security of the blockchain fundamentally vulnerable to things like human error, political disputes, or coordinated attacks. This centralization not only introduces many points of failure, but also poses risk of censorship and manipulation - trading the Long-Range-Attack vulnerability for many others.

Byzantine Fault Tolerance: HotstuffBFT

Byzantine Fault Tolerance (BFT) refers to a distributed systems' ability to reach a safe agreement on some arbitrary data when a bounded number of participants act faulty or maliciously, addressing the classic "Byzantine Generals Problem", where decentralized actors must coordinate to avoid catastrophic failure in the presence of traitors. Traditionally, BFT algorithms elect a leader among all distributed replicas to lead each view or round of consensus to completion. BFT is a fundamental prerequisite for decentralized networks and is the foundation of all modern blockchain consensus mechanisms.

In 2018, Cornell, UNC, and VMWare research published a paper [27] detailing Hotstuff BFT, "a leader-based Byzantine fault-tolerant replication protocol for the partially synchronous model". In this paper, the research team describes a highly optimized BFT implementation that has both only linear communication complexity and optimistic responsiveness. Over the past 6 years, Hotstuff BFT has been rigorously peer-reviewed, offering academically proven safety and liveness guarantees in partially synchronous (a peer-to-peer environment where the network messages may experience interruptions or delays) networks, with significant efficiency advantages over earlier protocols like PBFT and Tendermint BFT. While this paper introduced exciting concepts, the absence of a practical pacemaker and leader selection mechanism has led to Hotstuff BFT being regarded as a core academic reference rather than a production ready implementation.

Algorand Consensus: Sortition leader election

In 2017, Algorand network [28], pioneered by Silvio Micali and team, introduced a novel leader selection algorithm named 'Algorand Sortition'. At its core, Sortition was designed to be a distributed denial of service attack resistant solution, an area where competitors like Tendermint's round-robin algorithm fell short. The protocol works by each validator running a cryptographic lottery of sorts to determine their eligibility as a leader candidate. Fundamentally, the "lottery" uses a cryptographic primitive called a "verifiable random function" (VRF), which guarantees unpredictable, uniform distribution without bias. Because the process is private, potential leaders are not known until they announce their candidacy, mitigating the risk of a DDoS attack on the leader. Algorand Sortition introduced a significant innovation in leader selection, inspiring implementations like Polkadot's BABE, enhancing both the security and scalability of blockchain consensus mechanisms.

Verifiable Delay Functions: Chia Network

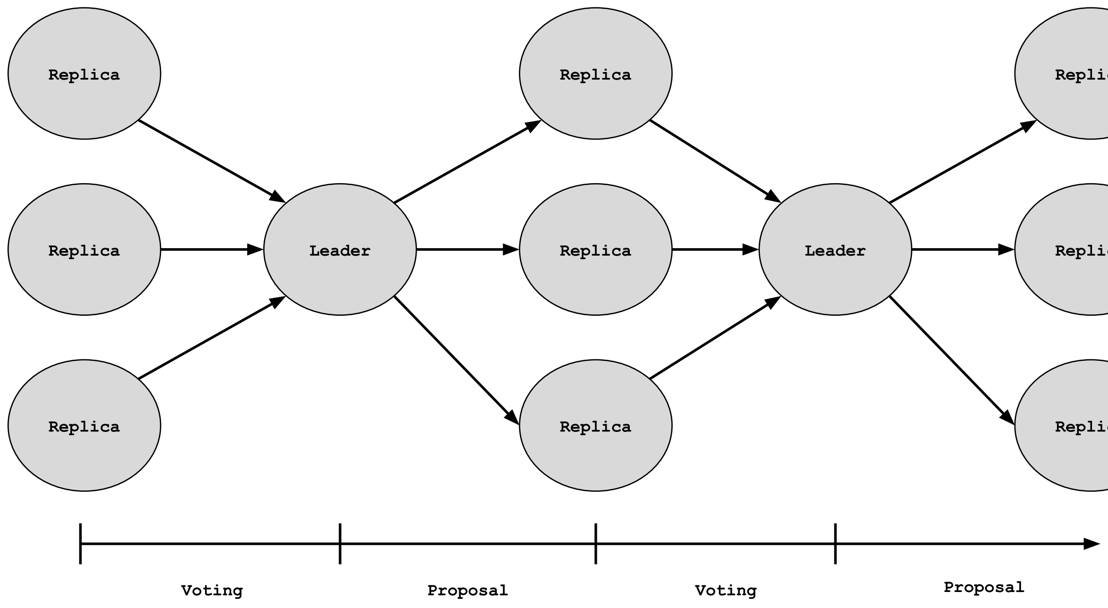
In 2021, Chia Network's [29] mainnet launched with a unique solution to Byzantine Faults, called Proof of Space and Time (PoST). PoST utilizes a cryptographic primitive called a "Verifiable Delay Function" which is designed to act as a reliable proxy of elapsed time. Unlike parallelizable hash computations used in Bitcoin's consensus algorithm, a VDF requires sequential computation, ensuring exponential advantage cannot be achieved with advanced or specialized hardware. Chia employs VDFs to create a predictable computational delay that ensures consistent blocktimes and prevents bias in the consensus mechanism. In recent years, VDFs gained popularity as a useful cryptographic construct for blockchain technology, valued for their ability to introduce trustless time-based operations.

6. Incubator Consensus

NestBFT: Fast, resilient and scalable consensus

Canopy Network presents NestBFT: an innovative consensus algorithm developed to (1) withstand grinding attacks while mitigating the risks of (2) DDoS and (3) long-range attacks. Unlike many modern consensus protocols that assume peer-to-peer network messages to arrive without delay - this protocol is designed for unreliable peer-to-peer environments. NestBFT is engineered with a novel pacemaker mechanism to address validator coordination challenges, while offering immediate safety and finality. The protocol is optimized to be highly efficient, using BLS multisignature aggregation for O(1) space complexity while utilizing a star communication pattern to maintain linear communication complexity. This design aims to be intuitive and straightforward to implement, as Canopy employs NestBFT for both the base-chain and sub-chain layers.

LINEAR STAR COMMUNICATION PATTERN



Leader Election: Simple sortition and fallback

NestBFT implements a DDoS resistant and highly available leader selection algorithm that is immune to grinding attacks. Drawing inspiration from Algorand's Sortition, the protocol uses a simplified form of a Verifiable Random Function (commonly known as a Practical VRF) in the form of a digital signature on seed data. The seed data consists of a fixed number of `last_proposer_addresses`, the current consensus `height` and `round`, ensuring unique input for each consensus view while being unable to be manipulated by a leader.

$$\text{Seed} = \text{Last\_Proposers\_Addresses} + \text{Height} + \text{Round}$$

$$\text{VRFOut} = \text{Hash}\left(\text{BLS\_Private\_Key.Sign(Seed)}\right)$$

Each validator executes the VRF each consensus round to produce a unique `'VRFOut'`. Unlike many existing protocols that rely on manipulable seed inputs like the `last_block_hash`, NestBFT's approach eliminates grindable bias, ensuring fairness and unpredictability. By combining simplicity with cryptographic integrity, this design enhances security and resilience in decentralized environments, offering robust protection against both DDoS and grinding attacks.

The next step of the election algorithm is for each validator to compute a numerical threshold that signals if a replica is a leader (block producer) candidate. This computation ensures stake amount proportionally weighs the chances of being the block producer. After this computation, the validator knows if they are a leader candidate by checking if their 'VRFOut' is below the threshold.

$$\text{CandidateCutoff} = \frac{\text{voting power} \times \text{expected candidates}}{\text{totalVotingPower}}$$

$$\text{IsCandidate} = \text{toNumberBetween0And1(VRFOut)} < \text{CandidateCutoff}$$

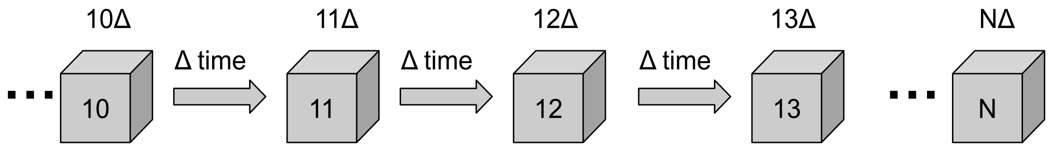
If deemed a candidate, the validator communicates the 'VRFOut' to all replicas during the first Election consensus phase. Multiple candidates are expected and the validator with the smallest VRFOut is chosen as the leader. In the case of zero candidates, the leader election algorithm falls back to a simple stake_weighted_random algorithm, which while being less DDoS resistant, is still immune to grinding attacks.

Algorithm 3 Fallback Leader Selection Algorithm

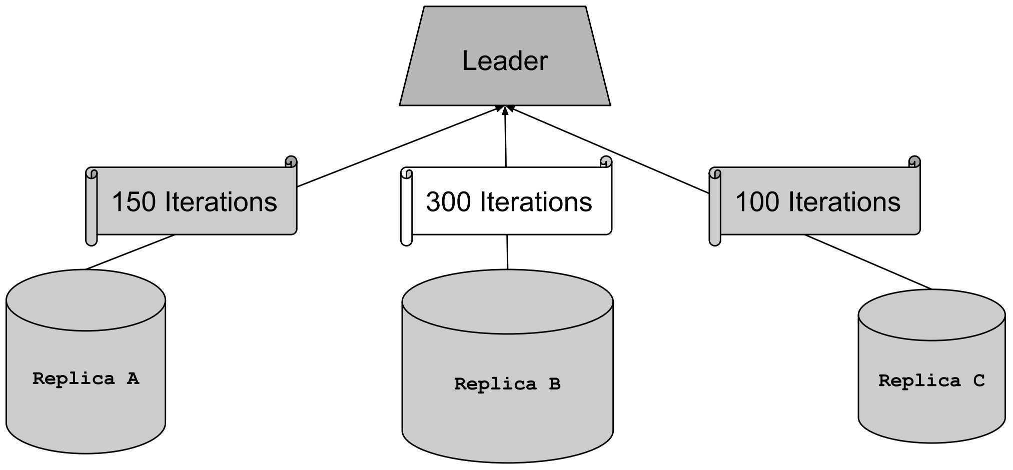
```
1: $\text{PowerIndex} \leftarrow \text{toNumber(Seed)} \bmod \text{validatorTotalTokens}$
2: $\text{index} \leftarrow 0$
3: for all $\text{Validator} \in \text{SortedVals}$ do
4:     $\text{index} \leftarrow \text{index} + \text{Validator.StakedTokens}$
5:     if $\text{index} > \text{PowerIndex}$ then
6:         return Validator
7:     end if
8: end for
```

VDFs: Verifiable mitigation of long-range attacks

Though the broader blockchain community accepts frequent checkpointing as a reasonable solution to prevent long-range-attacks, NestBFT strives to further eliminate reliance on centralized sources and social consensus by offering a novel approach. Given that the VDF primitive is a reliable proxy for elapsed time and is fundamentally resistant to specialized hardware, NestBFT employs Verifiable Delay Functions to ensure that the addition of each block to the chain is temporally consistent. By requiring a sequential proof of elapsed time in each block, the recreation of the blockchain by a long-range attacker is infeasible, as it would require both significant computational resources and time to reconstruct a chain that is longer than the network.



Here's how it works: each validator replica runs a VDF for just under the expected block time with seed data from the most recent block. The VDFs are aggregated by the producer during the `ELECTION_VOTE` phase of NestBFT and they include the VDF with the highest number of iterations in the proposal block. This design ensures the network gets the hardware benefit (if any) of the most computationally strong node in the network.



While this mechanism is effective in mitigating long-range-attacks, the priority of Canopy is sub-chain quality of life. Thus, by leveraging the long-range attack protection of the base-chain, Canopy offers checkpointing-as-a-service to sub-chains. Checkpointing information may be periodically included in a committee's `SUMMARY_TRANSCATION` and is indexed separately for each sub-chain, allowing full exportation of historical checkpoints upon sovereign graduation.

Phases: Step-by-step

This section explains the various phases of the consensus lifecycle, providing a thorough exploration of each stage to ensure a comprehensive understanding of how the consensus process works.

NestBFT Consensus Process

The consensus process of NestBFT is broken down into 8 core phases and 2 recovery phases. Each phase represents the smallest unit in the consensus process. Each round consists of multiple phases, and each height may consist of multiple rounds. These phases are executed sequentially to achieve consensus on the next block:

ELECTION

Each replica runs a Verifiable Random Function (VRF); if selected as a candidate, the replica sends its VRF output to the other replicas.

ELECTION-VOTE

Each replica sends ELECTION votes (signature) for the leader based on the lowest VRF value. If no candidates exist, the process falls back to a stake-weighted-pseudorandom selection.

PROPOSE

The leader collects ELECTION_VOTEs from +2/3 of the replicas, each including the lock, evidence, and signature from the sender. If a valid lock exists for the current height and meets Hotstuff's SAFE NODE PREDICATE, the leader uses that block as the proposal block. If no valid lock is found, the leader creates a new block to extend the blockchain. The leader then sends the new proposal (block, results, evidence) attaching the +2/3 signatures from ELECTION_VOTE to justify themselves as the Leader.

PROPOSE-VOTE

Each replica validates the PROPOSE message by verifying the aggregate signature, applying the proposal block against their state machine, and checking the header and results against what they produced. If valid, the replica sends a vote (signature) to the leader. Each vote vouches that the leader's proposal is valid.

PRECOMMIT

The leader collects PROPOSE_VOTEs from +2/3 of the replicas, each including a signature from the sender. The leader sends a PRECOMMIT message attaching +2/3 signatures from the PROPOSE_VOTE messages, justifying that +2/3 of the quorum believes the proposal is valid.

PRECOMMIT-VOTE

Each replica validates the PRECOMMIT message by verifying the aggregate signature. If valid, the replica sends a vote to the leader. Each vote vouches that the replica has seen evidence that +2/3 of the quorum believe the proposal is valid.

COMMIT

The leader collects PRECOMMIT_VOTEs from +2/3 of the replicas, each including a signature from the sender. The leader sends a COMMIT message attaching +2/3 signatures from the PRECOMMIT_VOTE messages, justifying that +2/3 of the quorum agree that a super-majority believes the proposal is valid.

COMMIT_PROCESS

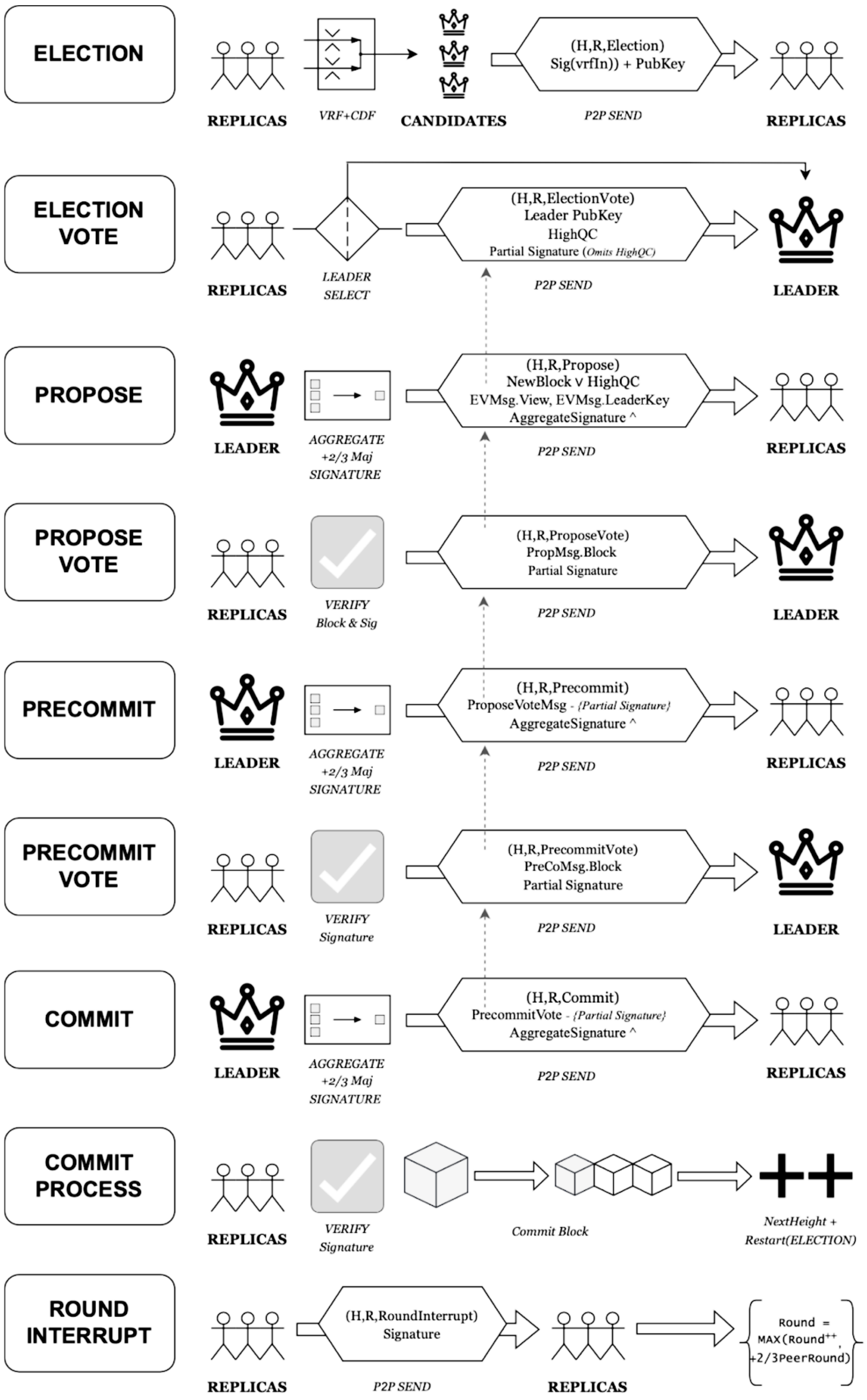
Each replica validates the COMMIT message by verifying the aggregate signature. If valid, the replica commits the block to finality, and resets the BFT for the next height.

ROUND_INTERRUPT (RECOVERY)

A failure in the BFT cycle caused a premature exit in the round. This results in a new round and an extended sleep time between phases to help alleviate any 'non-voter' issues. During this phase, each replica sends its View to all other replicas to alleviate round synchronization issues.

PACEMAKER (RECOVERY)

This phase follows ROUND_INTERRUPT. Each replica calculates the highest round a super-majority has seen and jumps to it to assist in 'round out of sync' issues.


ELECTION
REPLICAS
VRF+CDF
CANDIDATES
(H,R,Election)
Sig(vrfIn)) + PubKey
P2P SEND
REPLICAS
ELECTION VOTE
REPLICAS
LEADER SELECT
(H,R,ElectionVote)
Leader PubKey
HighQC
Partial Signature (Omits HighQC)
P2P SEND
LEADER
PROPOSE
LEADER
AGGREGATE +2/3 Maj SIGNATURE
(H,R,Propose)
NewBlock ∨ HighQC
EVMsg.View, EVMsg.LeaderKey
AggregateSignature ^
P2P SEND
REPLICAS
PROPOSE VOTE
REPLICAS
VERIFY Block & Sig
(H,R,ProposeVote)
PropMsg.Block
Partial Signature
P2P SEND
LEADER
PRECOMMIT
LEADER
AGGREGATE +2/3 Maj SIGNATURE
(H,R,Precommit)
ProposeVoteMsg - {Partial Signature}
AggregateSignature ^
P2P SEND
REPLICAS
PRECOMMIT VOTE
REPLICAS
VERIFY Signature
(H,R,PrecommitVote)
PreCoMsg.Block
Partial Signature
P2P SEND
LEADER
COMMIT
LEADER
AGGREGATE +2/3 Maj SIGNATURE
(H,R,Commit)
PrecommitVote - {Partial Signature}
AggregateSignature ^
P2P SEND
REPLICAS
COMMIT PROCESS
REPLICAS
VERIFY Signature
Commit Block
++
NextHeight + Restart(ELECTION)
ROUND INTERRUPT
REPLICAS
(H,R,RoundInterrupt)
Signature
P2P SEND
REPLICAS
Round = MAX(Round++, +2/3PeerRound)

Pacemaker: Optimal resync

NestBFT's novel pacemaker mechanism allows for optimal coordination in the case of an asynchronous networking event. Unlike legacy implementations like Tendermint, Canopy's pacemaker mechanism utilizes a communication phase to fast forward to the highest round that a supermajority of replicas have seen. During the PACEMAKER communication phase, the replicas gossip their current round - allowing an efficient calculation of the highest round that a supermajority of validator stake have witnessed, which then triggers a transition to that round. The waiting time between synchronization phases increases linearly with each round, ensuring that the system adapts to the network's speed while preventing unnecessary delays. The phase wait time is calculated as: phase_wait = round * initial_phase_Δ, allowing the system to gradually adjust to network conditions. The pacemaker combined with linearly increasing phase waiting time ensures a quick resynchronization of the validator set.

Preserving Finality: Life after independence

While applications have the option to stay on Canopy indefinitely, sub-chains running on Canopy are designed to be able to graduate to independence without interruption. Unlike any existing technology, Canopy provides a seamless framework for independence transition.

Here's how it works:

Algorithm 4 Sub-Chain Graduation to Independence

```
1: Input: Current sub-chain state with validator set and blocks
2: Output: Transition to independent blockchain with continuity
3: Sub-ecosystem selects a transition block to switch to the stand-alone version
   of Canopy
4: Sub-chain plugin is configured to sign new validator set as NextValidators
   in the final incubator block header
5: Continuity of both the validator set and blockchain are maintained
```

If the sub-chain is using the built-in `Checkpointing-as-a-Service`, it may easily export historical checkpoints and host them in a completely independent fashion. This ensures long-range attack protection by preserving the integrity and finality of the blockchain over time.

7. **Tokenomics**

CNPY: Overview

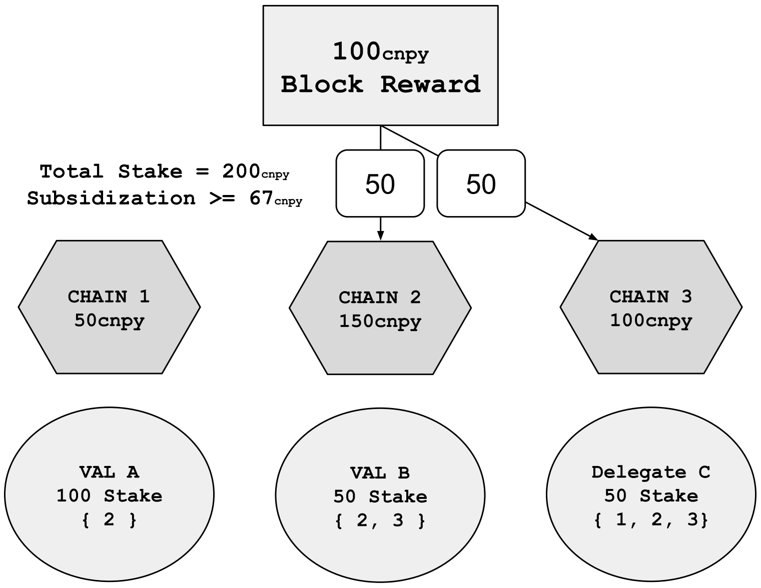
The native token CNPY has no pre-mint or pre-mine to follow fair launch principles. Similar to Bitcoin, Canopy has a fixed total supply with halvings that occur in regular intervals throughout the lifespan of the project to limit the total supply and reward actors within the protocol.

Supply: Minting and limits

Every block that is produced on Canopy Network creates new CNPY tokens. The beginning value of the block reward is [71,429] CNPY per block until it becomes effectively zero due to halvings. Block rewards are halved every [1,050,000] blocks or about [2] years. The total tokens is projected to be [21,000,000] CNPY.

Block Reward: Fair distribution

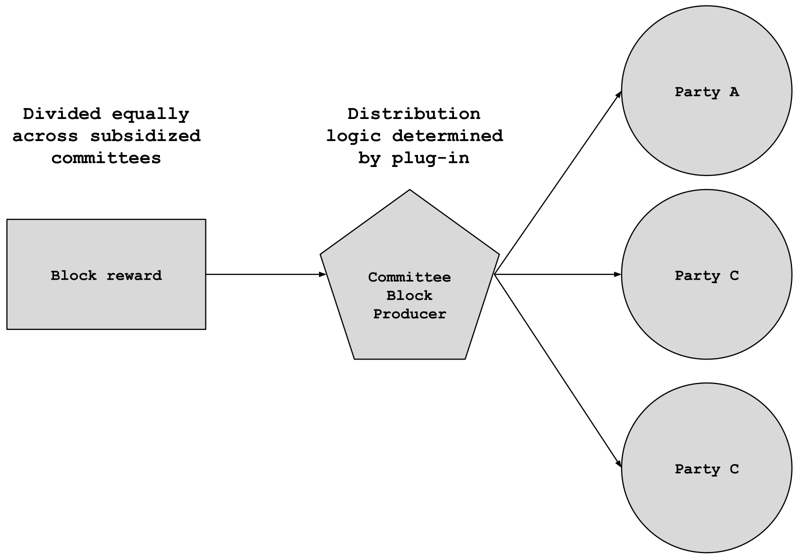
Each block, new CNPY is distributed evenly among `'subsidized'` sub-chain committees. A sub-chain committee becomes 'subsidized' once it surpasses a predefined threshold of total stake, which includes both validators' and delegates' stakes. Below is an illustrative example:



If a committee account is funded, a supermajority agreement from the committee members enables the transfer of those assets to many different recipients.

It's important to note that though the block reward is one way committee accounts are funded, the protocol supports a `SUBSIDIZE_TRANSACTION` that allows any address to transfer their assets to the committee to distribute at their discretion. It's worth noting that, unlike legacy projects, the Canopy base-chain committee operates under the same funding model as other sub-chains, making it virtually indistinguishable from them from an architectural perspective.

This flexible reward distribution model enables numerous unique applications. For example, a plugin for an external chain could reward validators of that chain in CNPY tokens based on the quality of their service, incentivizing cross-chain collaboration and performance. In contrast, the base-chain distribution model is straightforward - everything simply goes to the block producer and a delegator, both who are selected based on their stake amount relative to the other members of the committee.


Divided equally across subsidized committees
Distribution logic determined by plug-in
Block reward
Committee Block Producer
Party A
Party C
Party C

DAO: The Treasury Fund

The DAO fund is a treasury that is controlled by a supermajority of validators of Canopy. The DAO treasury fund receives a portion of the block reward, the default amount being [10%]. A supermajority of base-chain validators must agree to modify the default value.

Delegators: Influential, passive participation

CNPY holders that stake, but do not actively provide security services in committees are called delegators. Delegators stake to receive part of the block reward of that particular chain. Importantly, delegators have no slashing risk, which makes their role unique. Delegators play an important part in developer adoption, as they are the community driven gatekeepers of sub-chain subsidization status. Due to this, delegators are likely to receive a significant percentage of rewards from each committee, as their contributions are necessary to retain native funding of the committee reward pool.

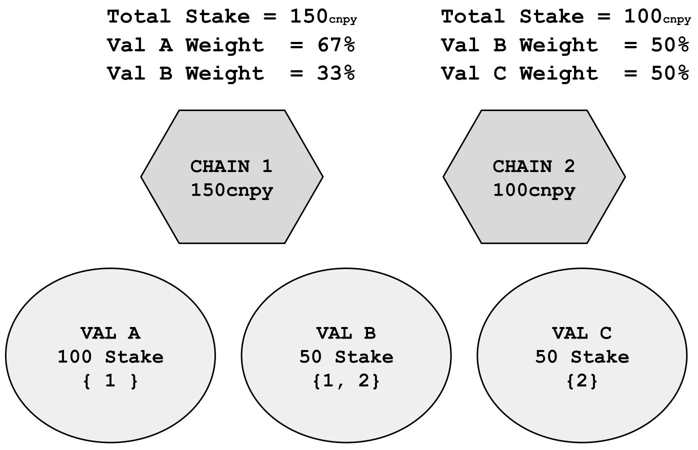
Restaking: Under the lens of Canopy

Canopy leverages proof of stake for its own security and to determine the subsidization status of sub-chains. When a validator or delegate submits a `STAKE_TRANSACTION` they bond tokens from their address as a surety bond against bad behavior. The more stake a validator or delegator holds relative to the other members, the higher their chances of being selected for token distribution. In the committee consensus, this means validators are more likely to be chosen as block producers, while delegators participate in a stake-weighted lottery for rewards.

In traditional staking, a validator's stake is only slashable for behaviors or actions performed within the blockchain where the validator has locked its tokens (the base-chain). However, Canopy restaking allows the total reuse of staked collateral from the base-chain to extend to multiple other sub-chains. This design enables stake to be used for security mechanisms and for subsidizing new projects - eliminating economic barriers to entry for sub-chains. This novel framework allows blockchain projects to launch with immediate economic security and have access to experienced node operators.

Since all of a validator or delegators stake is reappliable to whichever committees they stake for, the economics of each committee is straightforward. To determine the stake weight of an actor, simply take their stake amount divided by the total of the committee.

$$\text{total\_committee\_val\_stake} = \sum(\text{validator\_stake\_committed\_for\_this\_committee})$$
$$\text{validator\_stake\_weight} = \frac{\text{validator\_stake}}{\text{total\_committee\_val\_stake}}$$



Auto-compounding: Usability and incentive alignment

As a means of reducing the workload for stakers, rewards get added to the stake of the validator or delegator automatically. If auto-compounding is off, an early withdrawal penalty is taken from the reward and burned. This feature both simplifies the staking process and encourages long-term participation and alignment with the network's growth.

Other Incentives: The protocol after the block reward

Due to the halving mechanism, CNPY will eventually stop minting and native incentives for committee participation will cease. However, the incentive structure can continue through various means. The following sections detail this point, underscoring the longevity of the system.

Sub-Chain block reward: Aligning incentives

Unlike implementations like Avalanche, Canopy incentivizes committees to perform validation to ensure economic security is shared. However, sub-chains are encouraged to not rely on native base-chain incentives alone. Much like traditional staking L1s, a Canopy plugin may facilitate the distribution of a sub-chain token directly to validators (or other actors) in the form of a block reward. This time-tested approach creates direct alignment between the sub-chain and their committee. In order to remain competitive among other sub-chains, it is incumbent upon the sub-chain to determine appropriate additional sub-chain incentives to encourage the work and the staking. This mechanism also creates long-term sustainability for Canopy. As base-chain block rewards decrease over time, these rewards can be replaced by the sub-chain's, slowly weaning the ecosystem off of newly minted CNPY. This long-term handoff is a healthy transition for the ecosystem. Eventually, CNPY will be solely used to provide security or discretionary incentives by projects.

Subsidies: Community driven rewards

Chains supporters can choose to utilize the `SUBSIDIZE_TRANSACTION` on a particular sub-chain to contribute CNPY to reward validators for their work. They are designed to create long-term alignment between the chain and their validators. Depending on the implementation, the subsidy eliminates the need for recurring payments, creates financial certainty for the stakers of a given sub-chain, and incentivizes above and beyond the CNPY block reward (if that particular sub-chain is eligible). The subsidy transactions effectively create a two-sided marketplace where chains can pay validators for their block production - a healthy demand and supply relationship. Mechanically, it transfers tokens to the committee's reward pool - to be distributed as payments over a period of time set by the depositor. Using the **`OPCODE`** field of the transaction - subsidies are infinitely configurable to the needs of the chain. For instance, sub-chains could stack subsidies and have different distribution schedules that create interesting economics for the chain.

Subsidies may be configured in various strategies. Because Canopy reads and analyzes sub-chains, subsidies may be used to incentivize particular validators who meet certain criteria. This opens up opportunities for member chains to have increased control, flexibility, and creativity over validator rewards. As an example, a subsidy could be structured to be distributed only to validators that stake a certain amount of the chain's native token plus meet certain QoS requirements, creating a competition to stake more and have high QoS. Alternatively, the subsidy could be equal across all staked validators on a particular sub-chain. The combinations are endless and careful consideration must be given by member chains as to how to best incentivize Canopy validators.

A sub-chain must consider the following when enabling subsidies:

- The maximum rewards to be distributed.
- A time range for which the subsidy is valid and will be distributed.
- Any potential strategies or multipliers, which enables the chain to weigh the relative payout to each strategy within a single rewards submission (i.e. "distribute out to the top 10% of performers by QoS".)

8. Governance

Governance: The overview

Canopy is designed for effective on-chain governance and does not require the complexities that DAOs traditionally introduce. Canopy's governance structure utilizes validators as key decision makers in the system. Validators effectively act as elected representatives for sub-chains, serving until such a time as the validator pauses or unstakes itself from the system. Validators, though not "elected" in the traditional sense, hold tremendous responsibility and power by authorizing the inclusion of `PARAMETER_CHANGE` or `DAO_TRANSFER` transactions in blocks. Additionally, the incubator model includes built-in on-chain polling for any sub-chain. This mechanism allows real-time straw polling of both token-holders and validators for any internal or external plugin - providing a transparent and accessible view of community sentiment and decisions.

Proposal Transactions: Parameter Change and DAO Transfer

Parameters are on-chain configurations that influence the behaviors of the blockchain state-machine like `SEND_FEE` or (max) `BLOCK_SIZE`. These structures are built in knobs that allow the real-time modification of the base protocol without requiring a software upgrade or fork. To change the value of a parameter, a supermajority of committee members must agree on the new value of the parameter. If no agreement is reached, the value remains unchanged.

As described in the Tokenomics section of the paper, the governance treasury account receives a portion of the block reward. The funds are discretionary and may be transferred to any address based on a supermajority agreement of committee members. This structure enables the DAO to capitalize things like community-driven initiatives and ecosystem development.

Here's an illustration of how governance transactions work:

Governance Proposals and Transactions

1. Any persona may author a **`PARAMETER_CHANGE_TRANSACTION`** or **`DAO_TRANSFER`** using the built-in web wallet. These transactions have a start and end block window during which they are considered valid and may be submitted.

2. The author circulates the proposal throughout the community, communicating perspective and gathering support.

3. Validators may vote on this proposal using the web wallet, which populates a local JSON file in their data directory.
4. The author submits the transaction within the start and end block window.
5. Each validator processes the proposal according to its local JSON file, adding it to their mempool. A leader may then include the transaction in a block, with each replica voting based on their respective local files.
6. When the transaction is included in a final block, the proposal is immediately valid and applied at the same height.

Polling: Simple, adoptable, and extendable

Canopy enables sub-chains to participate in a decentralized polling platform without embedding complex voting features into their main chain. This approach prevents storage bloat and ensures the scalability of the sub-chain. Validator straw polling provides DAOs with real-time insight into voter sentiment on particular issues, enabling quicker decision-making. Additionally, Canopy's governance API collects voter metrics like power and actor type - helping sub-chains understand community sentiment before the actual voting takes place.

Polling through Canopy designed to be simple and easily adopted. Validators can cast their poll-votes by including a specifically formatted memo in any transaction within the designated voting period. These memos are then captured through chain analysis and factored into the polling results, ensuring efficient and transparent governance without additional burden.

Cross-Chain Governance: Multi-chain DAO

Traditional Web3 governance is siloed by chain and project, leaving constituents unable to vote on the issues of the broader ecosystem. Even in tightly bound ecosystems, governance can be isolated by chain and the chains do not have a say in the direction of the larger ecosystem. Isolation and silos do not need to be the norm and governance can grow to a body of sub-chains working together.

By interlinking validator committees across sub-chains, Canopy lays the groundwork for a multi-chain DAO. Think of it as a DAO of DAOs - a United Nations for L1s. Much like the United Nations or the World Economy, the value of such a system lies in the merging of interests between ecosystems and economies. As sub-chains are intertwined through the Canopy protocol, an interconnected governance structure is born. With a DAO of DAOs, unprecedented governance stability, security, and multi-protocol cooperation is created, especially for newer and emerging chains.

9. Concerns

Economic security with independence graduation

Since the blockchain incubator model heavily promotes independence graduation, there's a natural inclination to believe that long-term sustainability of the base-chain may be threatened by the successful exiting of sub-chains. Or that by prioritizing sovereignty, Canopy is not able to capitalize on the compounding value of the tokens within its ecosystem the way Ethereum has with its dApps. However, this logic does not consider the architectural implications of Canopy's tokenomics, which emphasizes sub-chain value capture throughout the entire lifecycle of the application. Whether a chain is subsidized or not, it is incumbent on that application to provide native token incentive to its committee and delegators to remain competitive and subsidized throughout its usage. Moreover, the flexible reward distribution model allows for continued tokenomic interaction and support even after a sub-chain graduates, allowing a sustained relationship with Canopy. Also, graduation from the incubator does not necessarily mean a complete departure from the Canopy ecosystem - as the sub-chain is still API compatible for plugin operations, thus maintaining the potential for ongoing collaboration. Furthermore, the validator set of the sub-chain are likely committee members for other sub-chains in canopy as well, making full exits of committee members unlikely.

Centralization of validators and cascading failures

Because the blockchain incubator model employs a shared validator set, there are concerns about increased centralization risks and the potential for cascading failures if one part of the system fails. The fear is that by having a single platform that supports the consensus and peer-to-peer layers for multiple sub-chains, the architecture is more inherently fragile than legacy projects. However, a comparable analysis of existing systems demonstrates this as unsubstantiated. For instance, Ethereum and Polkadot both utilize a single validator set to support their sub-chains (or dApps), the difference being that the blockchain incubator model does not enforce that every validator supports every sub-system - leading to a more resilient and flexible design.

Projects like Avalanche and Cosmos use separate validator sets for each sub-chain. Their designs do not actually support consensus or peer-to-peer layers for their sub-chains. Thus they do not address the economic and security adoption barriers that are fundamental to bootstrapping an L1. This weakness substantially impairs the value using the platforms.

While supporting sub-systems introduces additional risks, the benefits of addressing these core problems far outweigh these potential downsides. To mitigate the possibility of cascading failures, the protocol implements a safety eject feature to prevent committees who overlap members from suffering a significant loss in security due to the byzantine behaviors of validators or sub-chains. Canopy removes committee members who are slashed over a certain threshold from the committee - ensuring the protection of both parties from malicious or faulty actors. This mechanism safeguards the integrity of the network by isolating compromised participants before their actions can propagate further - eliminating real cascading risk.

Data Availability Layer

Popularized by the proliferation of Ethereum Rollups, data availability (DA) layer services are the latest trend in blockchain. In essence, DA layers abstract away the storage of sub-system's blockchains to a base-chain protocol like Celestia - promising enhanced scalability and availability. However, the blockchain incubator model outlines a novel abstraction that provides a more flexible modularization to preserve the ledger, while also being compatible with existing DA layers. Offloading or summarization of block data to another protocol cements a dependent relationship for the sub-chain, causing further reliance on the base-chain system without a defined path to reversibility. With DA layers, sub-chains are acting as lite-nodes to their own blockchain database, which can be both inefficient during access and a barrier for independence. Furthermore, modularizing persistence to another system may actually threaten scalability of the base-chain or be problematic for availability. This is because the maintainers of the data layer may not have natural incentives to preserve the ledger, requiring protocol-level mechanisms to ensure reliable storage.

FPGAs and VDFs

For long-range-attack mitigation, NestBFT relies on Verifiable Delay Functions of acting as a reliable proxy for elapsed time. A fundamental property of the cryptographic primitive is the property of being resistant to specialized hardware and parallelization. However, research conducted in 2019 by VDFAlliance [30] undercovered a nontrivial speedup of the Wesolowski VDF using customized algorithms applied to Field Programmable Array technology. VDFAlliance theorized a further speedup may be achieved using an application specific integrated circuit (ASIC), but has not published any news on their website since 2020. In order to mitigate this concern, NestBFT takes advantage of the fastest hardware on the base-chain committee. This means that even if only one committee member uses an FPGA or future ASIC, they would mitigate any hardware advantage an attacker may be able to have. In addition, Canopy Network will rely on social consensus as a fallback - likely implementing Checkpointing bi-annually, a vast improvement over legacy systems.

10. Conclusion and Comparables

Canopy Network: Revolutionizing Blockchain Ecosystem Development

Canopy Network's blockchain incubator model is the first progressive autonomy solution for blockchain ecosystems - offering the only comprehensive solution to the blockchain app lifecycle. In contrast to Avalanche and Cosmos, the Canopy protocol eases the required technical expertise and development challenges of running a sub-chain's consensus and peer-to-peer through a shared set of experienced node runners. With a unique consensus algorithm and flexible design, the model introduces novel security mechanisms for its sub-chains. The innovative restaking tokenomics creates unparalleled easy access for sub-chains - shifting service accessibility from financial investment to community popularity. Unlike Ethereum and Polkadot, the blockchain incubator is the only design that addresses both bootstrapping and provides a predefined track to L1 graduation, meanwhile providing frictionless upgrades during the app lifecycle. The unique plugin architecture enables unprecedented ease of interoperability, offering passive one-way integration for internal and external chains. By emphasizing simplicity and systemic understanding, Canopy Network positions itself as an educational leader, driving adoption to a broader audience and enabling a more accessible future for blockchain developers and enterprises.

Comparison Chart	Canopy	Ethereum	Polkadot	Avalanche	Cosmos
Offers immediate security for blockchain applications	✔	✔	✔		
Reduces financial & technical barriers for creating blockchains	✔	✔			
Provides a path to full sovereignty for apps	✔				✔
Interoperable with external ecosystems	✔				✔
Emphasizes architectural clarity	✔				
Validators are not required to secure all sub-chains	✔			✔	**N/A**
Sub-chains don't pay or stake to utilize the base service	✔				**N/A**
Simple application upgrades	✔			✔	**N/A**

11. References

1. *Nakamoto, Satoshi S. "Bitcoin: A peer-to-peer electronic cash system." SSRN Electronic Journal, 2008, https://doi.org/10.2139/ssrn.3440802.*
2. *Buterin, Vitalik. Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform, https://ethereum.org/content/whitepaper/whitepaper-pdf/Ethereum_Whitepaper_-_Buterin_2014.pdf.*
3. *Kwon, Jae. Tendermint: Consensus without Mining, https://tendermint.com/static/docs/tendermint.pdf.*
4. *King, Sunny, and Scott Nadal. PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake, 2012, https://www.peercoin.net/read/papers/peercoin-paper.pdf.*
5. *Kwon, Jae. Buchman, Ethan. "Internet of Blockchains." Cosmos Network, v1.cosmos.network/resources/whitepaper.*
6. *Polkadot-Io. "Polkadot-IO/Polkadot-White-Paper: The Technical Vision Paper for Polkadot, a Heterogeneous Extensible Multi-Chain." Polkadot.io, https://wiki.polkadot.network/docs/.*
7. *"Celestia White Paper." Celestia.Org, docs.celestia.org/.*
8. *"Avalanche Whitepaper & Documentation." Avax.Network, docs.avax.network/.*
9. *EigenLayer: The Restaking Collective, https://docs.eigenlayer.xyz/assets/files/EigenLayer_WhitePaper-88c47923ca0319870c611decd6e562ad.pdf*
10. *"Fundraising." Cryptorank.Io, cryptorank.io.*
11. *"Polkadot Timeline: A History of Polkadot." CryptoDaily, 2024, https://cryptodaily.co.uk/2024/06/polkadot-timeline-a-history-of-polkadot.*
12. *"Avalanche (blockchain platform)." Wikipedia, Wikimedia Foundation, https://en.wikipedia.org/wiki/Avalanche_%28blockchain_platform%29.*
13. *"The History of Solana." Medium, 21 Mar. 2021, https://medium.com/@tempestgirl1/the-history-of-solana-2ffd1f9b560e.*
14. *"Ethereum." Wikipedia, Wikimedia Foundation, https://en.wikipedia.org/wiki/Ethereum.*
15. *"Cosmos Milestones: Tendermint Core Evolution." Cryptopolitan, 2022, https://www.cryptopolitan.com/cosmos-milestones-tendermint-core-evolution.*
16. *Buterin, Vitalik. "Buterin Advocates for Censorship Tolerance in Special Cases." CryptoSlate, 7 Oct. 2022, https://cryptoslate.com/buterin-advocates-for-censorship-tolerance-in-special-cases.*
17. *ConsenSys. "The Inside Story of the CryptoKitties Congestion Crisis." ConsenSys, 18 Dec. 2017, https://consensys.io/blog/the-inside-story-of-the-cryptokitties-congestion-crisis.*

18. *Coin Metrics. "State of the Networks: Issue 281." Coin Metrics, 18 Nov. 2021, https://coinmetrics.substack.com/p/state-of-the-networks-issue-281.*
19. *"Decentraland Announces Layer-Two MANA Token Transfers, Plans for Full Migration." Cointelegraph, 10 May 2021, cointelegraph.com/news/decentraland-announces-layer-two-mana-token-transfers-plans-for-full-migration.*
20. *"Audius, the 'Decentralized Spotify,' Is Moving Part of Its Service to Solana Blockchain." CoinDesk, 29 Oct. 2020, https://www.coindesk.com/markets/2020/10/29/audius-the-decentralized-spotify-is-moving-part-of-its-service-to-solana-blockchain.*
21. *Celo Documentation. "Celo: The Layer 2 Solution." Celo, https://docs.celo.org/cel2#:~:text=As%20an%20L1%20chain%2C%20Celo,the%20vibrant%2C%20collaborative%20Ethereum%20community.*
22. *"The Ronin Migration." Axie Infinity Blog, https://blog.axieinfinity.com/p/migration.*
23. *Kain Warwick. "Why Optimism?" Synthetix Blog, https://blog.synthetix.io/why-optimism/.*
24. *Tether Official Website. Tether, https://tether.to/.*
25. *"A History of Uniswap." Uniswap Blog, https://blog.uniswap.org/uniswap-history.*
26. *Buterin, Vitalik. "Proof of Stake: How I Learned to Love Weak Subjectivity." Ethereum Foundation Blog, blog.ethereum.org/2014/11/25/proof-stake-learned-love-weak-subjectivity.*
27. *Abraham, Ittai. "HotStuff: BFT Consensus in the Lens of Blockchain." Arvix, https://doi.org/10.48550/arXiv.1803.05069.*
28. *Micali, Silvio. ALGORAND∗, 2017, https://26119259.fs1.hubspotusercontent-eu1.net/hubfs/26119259/Website-2024/PDFs/Algorand%20-%20Whitepaper.pdf.*
29. *"Chia Documentation." Chia Documentation, docs.chia.net/.*
30. *"VDF Research." VDF Research, vdfresearch.org/.*