CoSpaN: Permissionless Consensus in Sparse Networks

Thang N. Dinh[1], Jonathan Katz[2], Phuc D. Thai[3], and Hong-Sheng Zhou[1]

[1] Virginia Commonwealth University, Richmond, USA
{tndinh,hszhou}@vcu.edu
[2] Google, Washington DC, USA
jkatz2@gmail.com
[3] Sky Mavis, Ho Chi Minh City, Vietnam
phuc.thai@skymavis.com

Abstract. Security of blockchain systems depends on reliable dissemination, i.e., the ability of nodes to broadcast messages to all other nodes in the peer-to-peer (P2P) network. Reliable dissemination, assumed in many security analyses, does not always hold for real-world blockchains in which nodes do not connect to all but only a few other nodes. Indeed, various attacks such as eclipse and Sybil attacks are shown to disrupt the network, breaking the reliable dissemination assumption.

Several designs have been proposed to achieve reliable dissemination in a blockchain system, such as proof-of-stake (PoS). In PoS, the resources of nodes are known by everyone, and nodes randomly establish connections to other nodes based on their resources to construct expander graphs for propagating messages. However, these designs cannot be directly applied in proof-of-work (PoW) since there is no readily available proof of mining power. Moreover, the adversary can leverage the distribution of resources to identify highly connected nodes, i.e. nodes that control a significant amount of resources, and perform DoS attacks on those nodes in an attempt to disrupt the network.

In this paper, we present CoSpaN, the first network protocol designed for sparse networks in PoW setting. Nodes use merge-mining to provide proofs of mining power, and establish verifiable random connections such that the expected number of connections for each node is proportional to their mining power. Additionally, the CoSpaN protocol preserves the confidentiality of core nodes. This means that the adversary cannot determine whether or not a node is associated with a core node. As a result, the adversary is unable to launch DoS attacks to bring down core nodes and disrupt the network.

1 Introduction

The arrival of Bitcoin [23] marked the beginning of the blockchain era, promising a new decentralized economy without the risk of a single point of failure, monopoly, or censorship [5]. Bitcoin remains the most successful cryptocurrency despite the significant energy waste from computing resources. In Bitcoin [23], participants, called miners, use a *proof-of-work* (PoW) mechanism to generate new blocks: whoever can find a suitable nonce to solve the PoW is allowed to generate and appends to the best chain a new block. The newly generated block will be broadcast to all other miners via a peer-to-peer (P2P) overlay network.

1.1 Reliable dissemination

In a blockchain system, *Reliable dissemination* guarantees that if any valid message (e.g., blocks, transactions) is broadcasted by an honest node, it will be received by all other nodes within some bounded time. This is critical for consensus, e.g., provide the guarantee that newly generated blocks will be received by all nodes quickly. The existing analysis of Bitcoin security [14,26] assumes the availability of a network functionality that provides reliable dissemination. While this assumption can be achieved in a complete network, it may not hold in real-world P2P networks. These networks are typically sparse, i.e., each node have only a small number of connections. For instance, each node in the Bitcoin network establishes 8 outbound connections and accepts a maximum of 125 inbound connections [9,24]. Furthermore, nodes maintain the same set of connections over a long period. For example, observations in [4] indicate that 75% of Bitcoin nodes maintain the same set of connections even after 10 hours. To break the reliable dissemination assumption, several attacks have been shown in [17,31,21,15,25].

Eclipse attack on blockchain's P2P networks. The adversary aims to *eclipse* a node or a group of nodes by *monopolizing* all of their *inbound* and *outbound* connections, effectively isolating the target nodes from the rest of the network.

To *monopolize outbound connections*, the adversary can exploit the network protocol to lure honest nodes into establishing more outbound connections toward the adversary-controlled nodes [17]. Furthermore, in the permissionless setting, the adversary can spawn a large number of Sybil nodes, significantly increasing the chances for honest nodes to connect to adversary-controlled nodes.

The adversary can *monopolize inbound connections* in Bitcoin network through connection starvation attacks [10,17]. In this attack, the adversary employs a large number of (sybil) nodes to constantly request to establish connections with the target nodes, filling up all inbound connections of those nodes. Thus, the other honest nodes can hardly connect to the target nodes.

An impossibility against an adaptive adversary. In sparse networks, reliable dissemination is unachievable against an adaptive adversary that can *instantly* corrupt honest nodes and *known the network topology.* Specifically, consider a network with n nodes and an adversary who can dynamically corrupt any subset of f nodes. In theory, such an adversary can eclipse any node with fewer than $f+1$ connections by corrupting all of its neighbors [12,30,18].

Protocol	PoW	DoS resilience	Reliable dissemination	Verifiable random connection	Sparse network	Non-mining nodes
[20]	×	×	✓	×	×	×
[19]	×	×	✓	×	×	✓
[7]	×	×	×	✓	✓	×
CoSpaN	✓	✓	✓	✓	✓	✓

Table 1: A comparison between CoSpaN and existing designs for reliable dissemination.

Rigorous designs for reliable dissemination. There has been a growing interest in rigorous designs for reliable dissemination, as shown by recent works [22,20,7,19]. These designs are secure against a (weakly) adaptive adversary (refer to as a mildly adaptive or M-adaptive adversary) that needs to *wait for some time to corrupt* honest nodes.

In [20,19], nodes establish random connections based on their resources to form expander graphs, mitigating outbound monopolization. However, the lack of inbound connection verification leaves them susceptible to DoS attacks, where adversaries flood honest nodes with inbound requests. Limiting inbound connections, as in Bitcoin, helps but introduces vulnerability to connection starvation. Additionally, constructing a new random graph per message causes each node to eventually connect with most others—contradicting real-world P2P networks, where nodes maintain a small, stable set of peers over time.

The Generals' Scuttlebutt protocol [7] introduces a reliable dissemination design for proof-of-stake settings, where nodes use verifiable random functions (VRFs) to form connections based on their stake. This enables verifiable, stake-weighted connectivity, preventing adversaries from monopolizing either outbound or inbound links.

Challenges in PoW settings. Existing designs [20,7,19] assume that each node's resource level is publicly known. This assumption does not hold in PoW settings since there is *no readily available proof of mining power.*

Moreover, binding resources to physical node addresses is both unwarranted and risky. It allows adversaries to identify and target highly connected, resource-rich nodes with DoS attacks, potentially disrupting the network by disabling only a few critical nodes.

1.2 Our results

This work proposes a novel design, called CoSpaN, for reliable dissemination in PoW settings. Here, CoSpaN stands for **Co**nsensus in **Spa**rse **N**etworks.

Proof of mining power. Nodes use *merge-mining* to provide proofs of mining power, using the same PoW for generating new blocks, albeit with different thresholds. This way, the number of proofs of mining power is proportional to the mining power of nodes. After generating proofs of mining power, nodes submit them to the blockchain to make them readily available to everyone.

Based on proofs on mining power, nodes construct a bi-level random graph, called a *core-periphery graph*. Each proof of mining power entitles a node to act as a *core node* and establish connections to other nodes. Nodes that are not associated with any core nodes are referred to as *periphery nodes*. The core nodes select a set of neighbors, which includes both core and periphery nodes, to establish outbound connections. The selection of neighbors depends on the nodes' information and a random token extracted from the blockchain.

To ensure the connections are random and verifiable by the participating nodes of each connection, we design the functions to select the set of neighbors using inequalities over verifiable random functions (VRF) [11]. In addition, using VRF also ensures the confidentiality of connections, i.e., each connection is known only by the two participating nodes. This prevents adversaries from learning the network topology and allows the protocol to achieve reliable dissemination in a sparse network against an adaptive adversary. Shielding the network topology also provides protection against network disruption attacks such as BGP hijacks [29].

DoS resilience. To defend against DoS attacks on highly connected nodes, CoSpaN protocol preserves the confidentiality of core nodes by hiding the association between core nodes and their physical addresses. As a result the adversary cannot determine whether or not a node is associated with a core node. This prevents the adversary from launching DoS attacks to bring down core nodes and disrupt the network.

Security analysis in different threat models. Besides proving the security of CoSpaN protocol against the M-adaptive adversary, we also prove that the CoSpaN protocol is secure against a (weakly) *slow-observation* adaptive adversary (S-adaptive). In contrast with the M-adaptive adversary, the S-adaptive can *instantly corrupt* honest nodes but need to *wait for some time to discover a newly established connection.*

Comparision to existing designs. In Table 1, we compare our proposed CoSpaN protocol with existing designs for reliable dissemination.

PoW settings. The existing designs in [20,7,19] cannot be directly applied in PoW settings because there is no readily available proof of mining power. CoSpaN is the first design to provide reliable dissemination for sparse networks in PoW settings.

DoS resilience. In the designs in [20,7,19], the adversary could obtain the addresses of nodes with high connections by leveraging the distribution of resources and then perform DoS attacks on those nodes.

Reliable dissemination. The Generals' Scuttlebutt protocol [7] can only guarantee *partial* reliable dissemination. This means that a message broadcasted by an honest node will only be received by a large fraction of nodes in the network. In contrast, CoSpaN and the designs in [20,19] can achieve reliable dissemination.

Verifiable random connections. The designs in [20,19] lack a mechanism for verifying inbound connections, allowing the adversary to monopolizing the inbound connections of nodes. Meanwhile, CoSpaN and Generals' Scuttlebutt use VRF to allows the participant nodes to verify the connection.

Sparse networks. We say a network is sparse if each nodes in the network only connect with a few other nodes over a long period. In the designs in [20,19], nodes construct a distinct random graph for each message they send. Thus, over a long period, each node connects to most of the nodes in the network. On the other hand, the network in CoSpaN and Generals' Scuttlebutt protocol are sparse since nodes keep the same set of a few connections over a long period.

Non-mining nodes. The designs in [20,7] provide no connectivity to non-mining nodes. Those non-mining nodes, such as relay nodes and nodes that provide the tools and infrastructure for DApps, play an important role in blockchain ecosystems. In contrast, CoSpaN and the design in [19] provide connectivity guarantees for all nodes, including non-mining nodes.

1.3 Organization

We continue with the introduction of new models for consensus in sparse networks in Section 2. The design of CoSpaN protocol is presented in Section 3. The analysis of the security of CoSpaN against a static adversary and adaptive adversary are presented in Section 4 and Section 5, respectively. The numerical studies are shown in Section 6.

2 Consensus Models for Sparse Networks

In this section, we introduce our network-aware consensus model, which captures the *sparse* nature of real-world P2P networks. We further define the formal concept of *reliable dissemination* and the task of designing *consensus in sparse networks*, providing the first theoretical framework for analyzing consensus protocols in sparse networks.

2.1 Sparse network model (SNM)

We introduce a *sparse network model* (SNM) for a blockchain protocol execution. Extending the model by Pass, Seeman, and Shelat [26], referred to as the PSS model, our model includes a set N that consists of n ($n \in \mathbb{N}$) participating nodes, a blockchain protocol $(\mathit{\Pi}, \mathit{\Lambda})$, an environment $\mathcal{Z}$ that captures all aspects external to the protocol itself, and an adversary $\mathcal{A}$. The execution proceeds in *rounds* that model time steps.

There are three major differences between the SNM model and the PSS model, of which a summary of the PSS model is given in Appendix B.2. First, we consider a *point-to-point communication* in which nodes can only send messages after establishing connections to other nodes. This contrasts with the PSS model, which considers a communication model in which nodes can send messages to all other nodes. Secondly, we consider a *non-flat* mining model where each node can have a different amount of mining power, in contrast to the flat mining model (i.e., all nodes have the same mining power) in PSS. In particular, our non-flat model allows *nodes with zero-mining power*, e.g., nodes that participate in relaying messages but not the mining process. Such nodes and their corruption were not investigated in the PSS model. We are the first to *go beyond the standard corruption of the mining nodes and investigate the corruption of zero-mining power nodes that are responsible for communication functionality.* Finally, in the SNM model, we start with a *static adversary* that controls the same subset of nodes during protocol execution. Then, in Section 5, we consider two (weakly) *adaptive adversaries* who may corrupt honest nodes during the protocol execution but in certain restricted manners. [4]

Participating nodes. The set of participating nodes N can be partitioned into the set of *honest nodes* $\mathsf{H} \subset \mathsf{N}$, who strictly follow the blockchain protocol $(\mathit{\Pi}, \mathit{\Lambda})$, and the set of *malicious nodes* $\mathsf{N} \setminus \mathsf{H}$, which are controlled by an *adversary* $\mathcal{A}$. The number of malicious nodes $f = |\mathsf{N} \setminus \mathsf{H}|$ is bounded by $\eta \cdot n$ for a fixed $\eta \in (0, 1)$.

A blockchain protocol. Algorithm $\mathit{\Pi}$, which takes a security parameter κ as input, dictates how each node communicates and maintains a blockchain data structure $\mathcal{C}$, a collection of blocks. Algorithm $\mathit{\Lambda}$ contains a set of rules to extract a ledger $\mathcal{L}$ from $\mathcal{C}$, i.e., $\mathcal{L} = \mathit{\Lambda}(\mathcal{C})$. Here, the ledger $\mathcal{L}$ is a sequence of transactions, i.e., $\mathcal{L} = tx_1 \| tx_2 \| \ldots \| tx_i \| \ldots$. We also overload the notation $\mathit{\Lambda}$ to define the position of a transaction tx in the ledger. Specifically, for a transaction tx, $\mathit{\Lambda}(\mathcal{C}, tx) = i$ if $tx = tx_i$, the i-th transaction in the ledger, and $\mathit{\Lambda}(\mathcal{C}, tx) = 0$ if tx is not included in the ledger. The validity of a ledger is captured by a predicate V. The predicate $\mathsf{V}(\mathcal{L})$ returns 1 if and only if the ledger $\mathcal{L}$ is *valid* for some notion of validity (e.g., no double spending). The algorithm $\mathit{\Pi}$ is parameterized by V (denoted by $\mathit{\Pi}^{\mathsf{V}}$) to ensure that nodes always maintain valid ledgers.

[4] In the PSS model, nodes (with mining power) are allowed to be corrupted in an adaptive manner; furthermore, corrupted nodes can be uncorrupted. In our SNM model, both nodes with mining power and the ones with zero-mining power are allowed to be corrupted in weakly adaptive manners; in addition, already corrupted nodes are *not* allowed to be uncorrupted.

Non-flat mining model. All nodes interact with a random oracle $H : \{0,1\}^* \to \{0,1\}^\kappa$ (where κ is the security parameter) that takes an arbitrary length string and output a random value in $\{0,1\}^\kappa$. In each round, a node $i \in \mathsf{N}$ can make at most q_i queries to the random oracle H for some non-negative integer q_i. The integer q_i represents the node's "mining power". We denote $Q = \sum_{i \in \mathsf{N}} q_i$ as the total mining power of all nodes in N. The total mining power of all malicious nodes is at most $\rho \cdot Q$ where $\rho \in (0, 1/2)$ is the fraction of adversarial hash power. Further, the adversary can re-distribute the mining powers arbitrarily among the malicious nodes. This generalizes the "flat" model in [26] in which all nodes have the same mining power.

The non-flat mining model allows nodes with zero-mining power in the SNM model. If we consider the flat mining model, the fraction of malicious nodes η will equal the fraction of malicious mining powers ρ. As the adversary cannot control more than 50% of mining power, it cannot control more than 50% of nodes in the network. In the non-flat model, the adversary can control a large number of (sybil) nodes with zero mining power. Thus, the fraction of malicious nodes η can be arbitrarily close to 1.

Point-to-point communication. Any two nodes can establish a new connection or drop an existing connection. A node can only send messages to its neighbors, to whom it has established connections. These messages are guaranteed to be delivered to the neighbors after certain bounded number of rounds.
Establishing/dropping connections. It takes $t_\mathrm{e} \in \mathbb{N}$ rounds to establish a new connection between two nodes. At round r, two nodes start establishing a connection with each other. Then, at round $r + t_\mathrm{e}$, they can start exchanging messages. Once the connection is established, the nodes can instantly drop it.
Delivering messages. A node u only sends messages to a node v if there is a connection between u and v in that round. The adversary $\mathcal{A}$ is responsible for delivering all messages. The adversary cannot forge or alter the messages of honest nodes, but it can delay or reorder the messages as long as they are delivered within some *unknown delivery delay* $\delta \in \mathbb{N}$ rounds.
Connection observation. For the static adversary, we consider that newly established connections can be instantly observed by the adversary. In Section 5, we discuss the adaptive adversary and restrict their ability to observe connections. Specifically, the adversary can only observe new connections after a certain amount of time has passed.

A blockchain execution. The execution of the blockchain protocol is directed by the environment $\mathcal{Z}$, which initiates the set of nodes N. The nodes are initially designed as honest or malicious. We consider the execution with a polynomial in κ number of rounds. In each round, each node receives a list of transactions *txs* as input from the environment and adjusts its connections according to Π. Each honest node executes Π and incorporates any input and messages from its neighbors into its local data structure $\mathcal{C}$. Each node $i \in \mathsf{N}$ accesses the random oracle $H(\cdot)$ exactly q_i times with different nonces to find a valid proof of work (PoW). If an oracle call returns a proof of work, the node can generate a new block and send the new block to its neighbors, who will propagate it further.
Static adversary. We start with a static adversary $\mathcal{A}$ that controls the same subset of nodes throughout the entire execution of the protocol. In Section 5, we will analyze the security of our protocols against *(weakly) adaptive adversaries* that can corrupt more nodes during the protocol execution, but in certain restricted manners. To enable security in the presence of weakly adaptive adversaries, we will consider a restricted point-to-point communication model in which the adversary can only observe the connections among honest nodes after a sufficiently long time.

For adversary $\mathcal{A}$, and environment $\mathcal{Z}$, let $\mathsf{EXEC}_{\Pi^\mathsf{V},\mathcal{A},\mathcal{Z}}(\kappa)$ be a random variable denoting the joint view of all nodes (i.e., all their inputs, randomness, and messages received) over all rounds.

Admissible environments. We will be considering executions with restricted adversaries and environments. Given a predicate $\Gamma_\mathrm{stat}(\cdot,\cdot,\cdot,\cdot,\cdot)$, we define Γ_stat-admissible environment as follows.

Definition 1 (Γ_stat-admissible environments). *We say a tuple $(n, Q, \eta, \rho, \delta, \mathcal{A}, \mathcal{Z})$ with $\Gamma_\mathrm{stat}(n, Q, \eta, \rho, \delta) = 1$ is Γ_stat-admissible w.r.t $(\Pi^\mathsf{V}, \Lambda)$ if $\mathcal{A}$, $\mathcal{Z}$ are probabilistic polynomial-time algorithms, and for every* VIEW *in the support of $\mathsf{EXEC}_{\Pi^\mathsf{V},\mathcal{A},\mathcal{Z}}(\kappa)$, the following holds:*

1. *The adversary $\mathcal{A}$ can control at most $\eta \cdot n$ nodes, where n is the total number of nodes. The total mining power of the nodes controlled by the adversary is at most ρQ, where Q is the total mining power.*

2. *In every round r, the environment $\mathcal{Z}$ only sends the list of transactions txs as input to an honest player that maintains a ledger $\mathcal{L} = \Lambda(\mathcal{C})$ if the list of transactions can be appended to the ledger, i.e., $\mathsf{V}(\mathcal{L} \| txs) = 1$.*
3. *A message sent by an honest node is delayed by at most δ rounds before sending to its neighbors.*
4. *The adversary can instantly observe newly established connections among nodes.*

When context is clear, we refer $(n, Q, \eta, \rho, \delta, \mathcal{A}, \mathcal{Z})$ as Γ_{stat}-admissible.

2.2 Security Properties

We say a blockchain protocol is secure if it maintains a ledger that satisfies the *persistence* and *liveness* properties. The persistence property states that if a transaction is added to the ledger of an honest player, then it will be added to the same position in the ledgers of all honest players. The liveness property states that if a transaction is given as input to all honest players, the transaction should eventually appear on the ledgers of honest players. The security analysis [26] of a blockchain protocol relies on a property that any valid message can be disseminated from any honest node within a bounded time. We refer to this property as *reliable dissemination.*

Notations. Let $\mathcal{C}_i^r$ be the local chain of player i at round r. We define the persistence and liveness properties in the joint view $\mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa)$ as follows. We define $\textsc{view} \leftarrow \mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa)$ to denote that $\textsc{view}$ in the support of $\mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa)$. We define a function $\varepsilon(\cdot)$ is a *negligible* such that for any polynomial $\mathsf{poly}(\cdot)$, there exists some $\kappa_0 \in \mathbb{N}$ in which $\forall \kappa \geq \kappa_0, \varepsilon(\kappa) \leq \frac{1}{\mathsf{poly}(\kappa)}$.

Persistence. For a $\textsc{view} \leftarrow \mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa)$, we define $\mathsf{per}(\textsc{view}) = 1$ iff for any honest player i at round r and any transaction tx in the ledger $\Lambda(\mathcal{C}_i^r)$, there exists some $\Delta \in \mathbb{N}$ such that[5] any honest player j at round $r' \geq r + \Delta$, we have $\Lambda(\mathcal{C}_i^r, tx) = \Lambda(\mathcal{C}_j^{r'}, tx)$.

Definition 2 (Persistence). *A blockchain protocol* $(\Pi^{\mathsf{V}}, \Lambda)$ *achieves* persistence in Γ_{stat}-environments *if* $\Pr[\textsc{view} \leftarrow \mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa) : \mathsf{per}(\textsc{view}) = 1] \geq 1 - \varepsilon(\kappa)$.

Liveness. For $\textsc{view} \leftarrow \mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa)$ we define $\mathsf{liv}(\textsc{view}) = 1$ iff there exists a "wait time" parameter $\mathsf{t} = O(\kappa)$ such that for any transaction tx that is given as input for all the honest players from round r to round $r + \mathsf{t}$, we have, any honest player i at round $r + \mathsf{t}$, $\Lambda(tx, \mathcal{C}_i^{r+\mathsf{t}}) > 0$.

Definition 3 (Liveness). *A blockchain protocol* $(\Pi^{\mathsf{V}}, \Lambda)$ *achieves* liveness in Γ_{stat}-environments *if* $\Pr[\textsc{view} \leftarrow \mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa) : \mathsf{liv}(\textsc{view}) = 1] \geq 1 - \varepsilon(\kappa)$.

Reliable dissemination. The *reliable dissemination* property states that upon an honest node receives (or generates) a valid message, with high probability, the message will be disseminated to all (honest) nodes in the network within a bounded time. For a $\textsc{view} \leftarrow \mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa)$, consider a message msg that is first known by an honest player i at round r_{msg}. Here, the message msg could either be generated by the honest player i or sent from the adversary. For a parameter $\Delta \in \mathbb{N}$, we define $\mathsf{rel}(\textsc{view}, \Delta) = 1$ iff for any message msg that is known by an honest node at round r_{msg}, all honest nodes received the message msg at round $r_{msg} + \Delta$.

Definition 4 (Reliable dissemination). *A blockchain protocol* $(\Pi^{\mathsf{V}}, \Lambda)$ *achieves* Δ-reliable dissemination in Γ_{stat}-environments *if* $\Pr[\textsc{view} \leftarrow \mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa) : \mathsf{rel}(\textsc{view}, \Delta) = 1] \geq 1 - \varepsilon(\kappa)$.

The security analysis in Pass et al. [26] uses reliable dissemination[6] as an assumption. While this condition holds for fully connected networks, it may not necessarily hold for real-world blockchain networks, e.g., Bitcoin.

[5] In [26], Δ is network delay, i.e., the time it takes to propagate a message to all honest players.

[6] To be precise, the security analysis in Pass et al. [26] assume a *perfect reliable dissemination*, i.e., the messages from honest nodes will *always* be disseminated to all other honest nodes.

2.3 Consensus in Sparse Network

We now introduce the notion of network sparsity and then define the "consensus in sparse network" problem.

Network sparsity. The *sparsity* of a network is measured as *the average number of connections per honest node within a given period.* Consider a blockchain protocol execution, a VIEW in the support of $\mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa)$, and a parameter $t_{\mathrm{s}} = \Omega(\kappa)$. For a round r, let m_r be the number of connections that are established at some round $r' \in [r, r+t_{\mathrm{s}}]$ and have at least one end is an honest node. Let n_r be the number of nodes that are honest at round $r+t_{\mathrm{s}}$. We define the sparsity as $d_r = \frac{m_r}{n_r}$, the average number of connections per honest node from round r to round $r+t_{\mathrm{s}}$. For parameters d, t_{s}, we define $\mathsf{spa}(\textsc{view}, d, t_{\mathrm{s}}) = 1$ iff for any round r in the execution VIEW, we have, $d_r \leq d$.

Definition 5 (Network sparsity). *A blockchain protocol* $(\Pi^{\mathsf{V}}, \Lambda)$ *achieves* d-sparsity in Γ_{stat}-environments *if there exists a parameter* $t_{\mathrm{s}} = \Omega(\kappa)$ *such that* $\Pr[\textsc{view} \leftarrow \mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa) : \mathsf{spa}(\textsc{view}, d, t_{\mathrm{s}}) = 1] \geq 1 - \varepsilon(\kappa)$.

We define the consensus in sparse network as follows.

Definition 6 (Consensus in Sparse Network). *The goal is to construct a consensus protocol* $(\Pi^{\mathsf{V}}, \Lambda)$ *in a* Γ_{stat}*-admissible environments that can achieve together 1)* persistence *and* liveness*; and 2)* sparsity.

3 Protocol Design

The CoSpaN protocol extends Nakamoto's protocol [23] and implements a P2P network protocol that provides reliable dissemination.

In CoSpaN protocol, nodes form a new network topology every epoch, which is a duration of L consecutive blocks. During an epoch, nodes use merge-mining to generate readily available proof-of-mining power and establish pseudo-identities called *core registrations*, which include public keys and a commitment to their network addresses. Each registration entitles a node to act as a *core node* in the next epoch. Thus, a node with high mining power is often associated with multiple (logical) core nodes. Nodes that are not associated with any core nodes are referred to as *periphery nodes.*

The main goal of CoSpaN protocol is to construct a *core-periphery topology* that connects all honest nodes in a small-diameter network. To achieve this, nodes are required to establish *verifiable random connections* with each other. Before accepting a connection, nodes verify its validity using a combination of a verifiable random function (VRF) [11] and blockchain-extracted randomness. This lightweight verification is effective in preventing the adversary from monopolizing connections from an honest node and performing eclipse attacks. It also makes the connections between honest nodes unpredictable, thereby making *connections private* and each one is known only by the two participating nodes.

In our protocol, we preserve the confidentiality of core nodes by hiding the association between core nodes and their physical addresses. Core nodes play a crucial role in maintaining network connectivity. At a high level, the set of core nodes can be viewed as a decentralized, well-connected hub that connects all the peripheral nodes. However, this also presents a new vector of attack, where adversaries attempt to perform DoS attacks to bring down as many core nodes as possible. If a large enough number of core nodes are attacked, this can disrupt the network. By preserving confidentiality of core nodes, we can prevent such attacks from occurring.

3.1 Protocol design

Protocol Π_{CSN} employs an *epoch-based reconfiguration*, where a new topology is constructed every epoch. The reconfiguration in each epoch is divided into two stages: *core selection* and *topology construction.* The pseudocode of protocol Π_{CSN} can be found in Fig. 1.

CoSpaN Protocol Π_{CSN}

Parameters:

- Epoch length L, grace period half-length $k = \Omega(\kappa)$
- Core width s, connection parameter d

Local state: Node u has a state $\langle \mathcal{C}, \mathsf{ip} \rangle$ containing blockchain copy $\mathcal{C}$ and its network address ip.

```
Compute block height $l = len(\mathcal{C})$
Compute epoch number $i = \lfloor l/L \rfloor + 1$
**Core selection:** If $l < L - 2k$
  Generate a pseudo-identity $\mathsf{cid}$
  Set $\mathsf{context} := \langle \mathsf{prev}, MkRoot(txs), \mathsf{cid} \rangle$
  Upon finding a *nonce* such that
      $H(\mathsf{context}, nonce) \in [\mathtt{T}, \mathtt{T} + \mathtt{T}_{\text{core}}]$ (Eq. 2),
  propagates a core registration
      $\mathsf{cr} = \langle \mathsf{context}, nonce \rangle$.
**Topology construction:** If $l \in [iL - k, iL)$
*Randomness extraction:*
**If** $l = iL - k$, concatenate all the blocks with block heights in $[iL - 3k, iL - 2k)$ and return the hash value as $\mathsf{rnd}_i$

*Verifiable random connections:*
Let $\bar{\mathsf{C}} = \{$core registrations in the registration period$\}$
Let $\bar{\mathsf{C}}_u \subseteq \bar{\mathsf{C}}$ be the set of core registrations from node $u$
Core-core connections:
**For** each core registration $\mathsf{cr} \in \bar{\mathsf{C}}_u$ **do**
  – *Step 1: Announcing eligible connections*
    **For** each core registration $v \in \bar{\mathsf{C}} \setminus \bar{\mathsf{C}}_u$ **do**
      With probability $\bar{\mu}$, announce the address to $v$ via a eligible message.
  – *Step 2: Establishing connections*
    **For** each eligible message from $w$ with the address $w.\mathsf{ip}$
      Connect to $w.\mathsf{ip}$ after verification.
  – *Step 3: Accepting connections*
    **For** each connection request
      Accept the connection after verification.

Core-periphery connections:
*[u as a core node] Establishing connections*
**For** each core registration $\mathsf{cr} \in \bar{\mathsf{C}}_u$ **do**
  **For** each periphery node $v \neq u$ with address $v.\mathsf{ip}$ **do**
    With probability $\bar{\mu}$, connect to $v.\mathsf{ip}$.
*[u as a periphery node] Accepting connections*
**For** each connection request from $w$
  Accept the connection after verification.
```

Fig. 1: CoSpaN protocol Π_{CSN}.

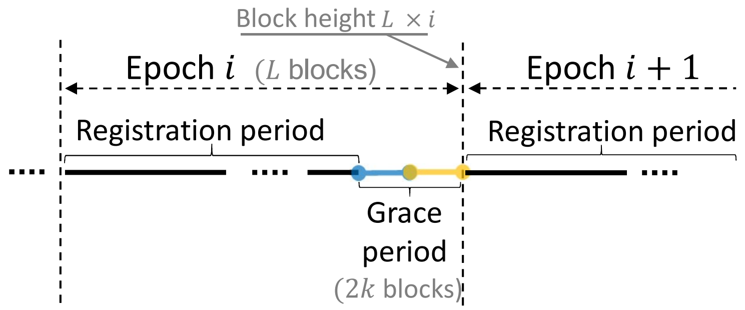


Fig. 2: Epoch of L blocks consisting of 1) a registration period in which nodes use a PoW merged-mining to perform core registration and 2) a short grace period of $2k$ blocks for $k = O(\kappa)$. The first half of the grace period is to guarantee the convergence of nodes' views on the set of core registrations. In the second half of the grace period, nodes will construct a new network topology using verifiable random connections, computed from the core registrations.

Epoch-based Configuration As shown in Fig. 2, each epoch spans L consecutive blocks, where $L \in \mathbb{N}$ is a fixed parameter. For a node, the ith epoch includes rounds with block heights in $[(i-1)L, iL)$, where block height is the length of the longest valid chain observed. Due to network delays and adversarial behavior, block heights may vary across nodes.

Each epoch is divided into two smaller periods: a *registration period* for core selection, followed by a short *grace period* for topology construction. For $k = \Omega(\kappa)$ and $k \ll L$, we set the lengths of the registration and grace periods to $L_{\mathtt{reg}} = L - 2k$ and $2k$, respectively. The grace period serves to tolerate any potentially different views of the nodes. During the first half of the grace period, consisting of k blocks, there is sufficient time to rule out any divergence among nodes regarding the set of core registrations, i.e., the set of core nodes in the next epoch. Similarly, the k blocks in the second half allow sufficient time for nodes to establish new connections. During the grace period, the nodes maintain their existing connections and only drop them when the new epoch begins to ensure network connectivity continuity.

Core selection During the registration period, the nodes are selected via the same PoW mining to find new blocks, albeit, with different thresholds [28,2].

To generate new blocks, nodes attempt to solve a PoW puzzle by searching for a *nonce* over a $\mathsf{context}$ that satisfies the hash inequality

$$H(\mathsf{context}, \mathit{nonce}) < \mathtt{T},$$

where $\mathtt{T}$ is the mining difficulty.

The context consists of a pointer (hash value) prev to the previous block, the Merkle root $\mathit{MkRoot}(txs)$, a single hash value to validate the included transactions, and a network pseudo-identity cid.

$$\mathsf{context} = \langle \mathsf{prev}, \mathit{MkRoot}(txs), \mathsf{cid} \rangle. \tag{1}$$

Whenever the node finds a *nonce* that

$$\mathtt{T} \leq H(\mathsf{context}, \mathit{nonce}) < \mathtt{T} + \mathtt{T}_{\mathrm{core}}, \tag{2}$$

where $\mathtt{T}_{\mathrm{core}}$ denotes *core registration difficulty*, it generates and propagates a core registration

$$\mathsf{cr} = \langle \mathsf{context}, \mathit{nonce} \rangle.$$

The core registrations will be propagated within the P2P network and included in the ledger as (prioritized) transactions.

Denote by s the *core width*, which represents the expected number of core registrations (i.e. the number of core nodes) in each epoch. To ensure $\Theta(s)$ core registrations in each period, we set $\mathtt{T}_{\mathrm{core}} = \frac{s}{L_{\mathtt{reg}}}\mathtt{T}$. This can be verified by noting that the mining difficulty T leads to approximately $L_{\mathtt{reg}}$ blocks during the registration period. Our later security analysis remains valid when the number of actual registrations is within a constant ω, such as $\omega = 90\%$, of the core width s. This adds robustness to the protocol, for example against low participation of nodes in the core selection.

Core-periphery topology construction During the second half of the grace period, the *core-periphery topology* is constructed in two phases. First, all nodes in the network *extract randomness* from the previous epoch. Then, based on this randomness, core nodes establish *verifiable random connections* to other core and periphery nodes.

The probability that a core node establishes a connection to another (core or periphery) node is $\bar{\mu} = \frac{d}{2s}$, where d is the *connection parameter* and s is the *core width*. Note that the expected number of (logical) core nodes associated with the same (physical) node (or miner) i is proportional to its mining power q_i, as is the expected number of connections established by i.

Randomness extraction. Similar to the method described in [8], nNodes compute the randomness rnd_i as the hash value of the concatenation of the last k blocks in the registration period, which have block heights in the range $[iL - 3k, iL - 2k)$. This ensures that all nodes obtain the same randomness rnd_i at the second half of the grace period.

Connection establishment. During the second half of the grace period, all nodes utilize core registrations and extracted randomness to establish new connections. The connection establishment procedure is divided into two sub-procedures for two types of connections: *core-core connections* and *core-periphery connections.*

- Core-core connections. Each core node u establishes a connection with every other core node v with a probability of $\bar{\mu}$. Two core nodes u and v are connected if either u establishes a connection to v or v establishes a connection to u.
- Core-periphery connections. Each core node u establishes a connection with every periphery node v with a probability $\bar{\mu}$. Periphery nodes do not establish connections with other nodes.

3.2 Security improvements

We describe how nodes establish verifiable random connections to construct the core-periphery graph while preserving core anonymity.

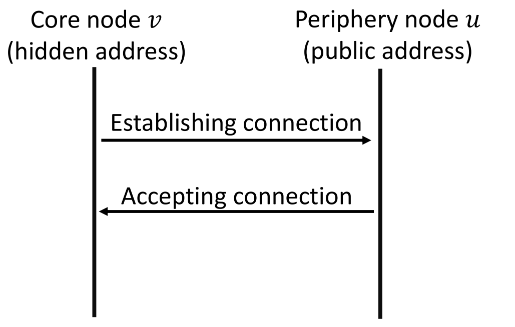
Verifiable random connections To establish verifiable random connections, a public-key pseudorandom function known as a VRF is employed [11]. The VRF provides proofs that its outputs were calculated correctly and randomly, making them difficult to predict. The owner of the secret key can compute the function value σ and an associated proof π for a given input value. Others can use an algorithm called Verify, along with the proof and the associated public key, to check if the function value was calculated correctly without revealing any information about the secret key. For the formal definition of the VRF, please refer to Appendix A.2.

Consider a core node u that hold the key pair $(\mathsf{sk}', \mathsf{pk}')$ for VRF. For a (core or periphery) node v, let id_v be the identity of node v. Here, if v is a core node, we set the identity id_v as the pseudo-identity cid_v of v. If v is a periphery node, we set the identity id_v as the address ip_v of v. The node u computes $(\sigma_v, \pi_v) := \mathsf{Prove}_{\mathsf{sk}'}(\mathsf{rnd}_i || \mathsf{id}_v)$. We say that u is *eligible* to connect to v, in epoch $i+1$, if and only if output σ_v is smaller than a threshold $\mathtt{T}_{\text{con}} = \bar{\mu} \cdot 2^{\kappa}$, i.e.,

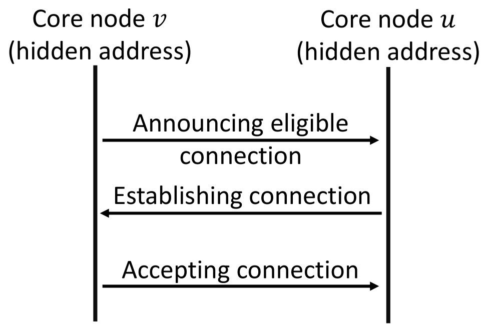
$$\sigma_v < \mathtt{T}_{\text{con}}, \tag{3}$$

For each node v, the probability that the Eq. 3 is satisfied and node u is eligible to connect to node v is $\bar{\mu}$. Upon receiving a connection request from u, the node v can verify node u is eligible by verifying 1) (σ_v, π_v) is computed correctly, and 2) $\sigma_v < \mathtt{T}_{\text{con}}$.

Confidentiality of core nodes As we mentioned, we preserve the confidentiality of core nodes by hiding the association between core nodes and their physical addresses. Thus, the node's address is not included in the pseudo-identity to preserve anonymity. To generate cid, a node generates key pairs $(\mathsf{sk}', \mathsf{pk}')$, $(\mathsf{sk}, \mathsf{pk})$, $(\bar{\mathsf{sk}}, \bar{\mathsf{pk}})$ for a VRF, a public-key encryption scheme, and a signature scheme, respectively. The network pseudo-identity is returned as $\mathsf{cid} = \langle \mathsf{pk}', \mathsf{pk}, \bar{\mathsf{pk}} \rangle$.



(a) Core-periphery connection.



(b) Core-core connection.

Fig. 3: Establishing connections. To preserve core anonymity, an extra step is added to establish core-core connections.

In Fig. 3, we show the interactions to establish core-periphery and core-core connections. The establishment of a core-periphery connection consists of two steps. First, a core node u requests to connect to a periphery node v if it is eligible to do so. Then, node v will accept the connection after verifying that u is eligible.

For the core-core connections, as the addresses of core nodes cannot be obtained from the pseudo-identities, an extra step is added to establish core-core connections. First, each core node u need to broadcast an encryption of it address to the core nodes that it is eligible to connect. Node u creates a message M containing information on eligible connections to broadcast to all nodes in the P2P network. For each "eligible" core node v, u uses v's public key, $v.\mathsf{pk}$, for encryption. It computes $\xi \leftarrow \mathsf{Enc}_{v.\mathsf{pk}}(\sigma_{\text{core}}, \pi_{\text{core}}, \mathsf{ip})$ and adds ξ to a list M. Node u then computes a signature $\gamma = \textsc{Sign}_{\bar{\mathsf{sk}}}(M, \mathsf{cr})$ and broadcasts (M, cr, γ) to all nodes in the network. By using encryption, u limits the visibility of its network address, $u.\mathsf{ip}$, to only those nodes that u is eligible to connect to in the future. To prevent spamming attacks, every node will only forward at most one valid message (M, cr, γ) for each core registration cr. Then, the core node v can decrypt the message to extract the address of the node u. After verifying u eligible, node v request to make connection to node u, by sending a signature $\textsc{Sign}_{\bar{\mathsf{sk}}}(v.\mathsf{cr})$. The node u then accept the connection if the signature of node v is correct.

3.3 Complexity

Computation cost. On average, each core node performs 1) $n + s$ times to compute the VRF (each for a core or periphery node); 2) d times to encrypt (each for an eligible core node); and 3) ds times to decrypt (for every encryption from core nodes). In total, the time complexity of a core node in each epoch is $O(n + d \cdot s)$. On average, each periphery node performs d times to verify the connection. Thus, the time complexity of a periphery node in each epoch is $O(d)$.

Communication cost. In an epoch, each core node broadcasts a message that contains the encryption to all other node. The message complexity to broadcast the encryption is $O(n \cdot s)$. The remaining communication happens between the nodes that are connected to each other in that epoch. The message complexity to this communication is $O(n + s)d$. Thus, the total message complexity in each epoch is $O(n \cdot s + n \cdot d + s \cdot d)$.

4 Security analysis

Threat model	Adaptive corruption	Connection observability	Epoch length L	Core width s	Connection parameter d
Static	n/a	Instantly	$\Omega\left(\frac{\kappa}{(1-\rho)\omega}\right)$	$\Omega\left(\frac{\kappa}{1-\rho}\right)$	$O\left(\frac{\kappa}{-\log((1-\rho)\omega)}+\log s\right)$
M-adaptive	τ_{corr} rounds	Instantly	$O\left(\tau_{\text{corr}}\cdot\alpha\right)$	$\Omega\left(\frac{\kappa}{1-\rho}\right)$	$O\left(\frac{\kappa}{-\log((1-\rho)\omega)}+\log s\right)$
S-adaptive	Instantly	τ_{obs} rounds	$O\left(\tau_{\text{obs}}\cdot\alpha\right)$	$\Omega\left(\frac{\kappa+h}{1-2\rho}\right)$	$O\left(\frac{\kappa}{-\log((1-2\rho)\omega)}+\log s\right)$
S-adaptive$\langle\phi,\gamma\rangle$	Instantly	τ_{obs} rounds	$O\left(\tau_{\text{obs}}\cdot\alpha\right)$	$\Omega\left(\frac{\kappa+h\cdot\phi}{\gamma-\rho}\right)$	$O\left(\frac{\kappa}{-\log((\gamma-\rho)\omega)}+\log s\right)$

Table 2: Feasible parameter settings of CoSpaN (L, s, d) against different adversary models. CoSpaN, parameterized by the epoch length L, the core width s, and the connection parameter d can achieve reliable dissemination with a sparsity of $\frac{2}{1-\eta}(1+\frac{s}{n})d = O(\log n)$, where n is the number of nodes and η denotes the fraction of malicious nodes. Defending against (weakly) adaptive adversaries requires (1) shorter epoch lengths and (2) larger numbers of core nodes to limit the effect of targeted corruption toward the core nodes. In an S-adaptive$\langle\phi,\gamma\rangle$ model, where $\phi \in (0,1)$ and $\gamma \in (\rho, 1-\rho)$), we consider an S-adaptive adversary when the top ϕ fraction of honest nodes control a fraction γ of mining power.

We prove that by choosing sufficient parameters (Theorem 4.3), the CoSpaN protocol can achieve security properties (defined in Section 2.2) under the presence of a static adversary in a network with $O(\log n)$ sparsity. Later, in Section 5, we will show that the CoSpaN protocol can also achieve security properties under the presence of weakly adaptive adversaries. In Table 2, we provide a summary of the security analysis of the CoSpaN protocol against a static adversary and two additional weakly adaptive adversaries. An M-adaptive adversary needs to *wait for τ_{corr} rounds to corrupt an honest node.* An S-adaptive adversary can corrupt a node instantly, however, needs to *wait for τ_{obs} rounds to discover a newly established connection.* We also consider an S-adaptive$\langle\phi,\gamma\rangle$ setting, in which the top ϕ fraction of honest nodes control a fraction γ of mining power. The S-adaptive setting can be seen as a special case of S-adaptive$\langle\phi,\gamma\rangle$ with $\phi = 1$ and $\gamma = 1-\rho$.

4.1 Security properties

We prove that protocol Π_{CSN} achieves security properties (including persistence and liveness) through *induction* (see Fig. 4). In each induction step, we assume reliable dissemination in the current epoch and prove that protocol Π_{CSN} achieves security properties that lead to reliable dissemination in the next epoch. For the basis step, we assume a *bootstrapping condition* that provides reliable dissemination for protocol Π_{CSN} in the first epoch.

We prove that the Π_{CSN} protocol achieves the necessary security properties for the ledger (persistence and liveness) and consensus (chain growth, chain quality, and consistency). For simplicity, we will refer to both the security properties for the ledger and consensus as the security properties for the remainder of this paper. For a given $\text{VIEW} \leftarrow \mathsf{EXEC}_{\Pi^{\text{V}},\mathcal{A},\mathcal{Z}}(\kappa)$, we define $\mathsf{sec}(\text{VIEW}) = 1$ if the following conditions are met.

- *Persistence and liveness.* $\mathsf{per}(\text{VIEW}) = 1$ and $\mathsf{liv}(\text{VIEW}, \mathsf{t}) = 1$ (see Subsection 2.2).
- *Chain growth.* There exists a parameter $\mathsf{cg} > 0$ such that, for any honest player i at any rounds r and $r' = r + \Omega(\kappa)$, we have, $\mathsf{len}(\mathcal{C}_i^{r'}) - \mathsf{len}(\mathcal{C}_i^{r}) \geq (r'-r)\mathsf{cg}$, where, $\mathcal{C}_i^{r}, \mathcal{C}_i^{r'}$ are the chains of the player i at round r, r', respectively.
- *Chain quality.* There exists a parameter $\mathsf{cq} > 0$ such that, for any honest player i at any rounds r, among any $k = \Omega(\kappa)$ consecutive blocks in the chain $\mathcal{C}_i^{r}$, the fraction of honest blocks is at least cq.
- *Consistency.* For any honest player i at any rounds r and any honest player j at any rounds $r' > r$, after removing the last $k = \Omega(\kappa)$ block of $\mathcal{C}_i^{r}$, we obtain a prefix of the chain $\mathcal{C}_j^{r'}$.

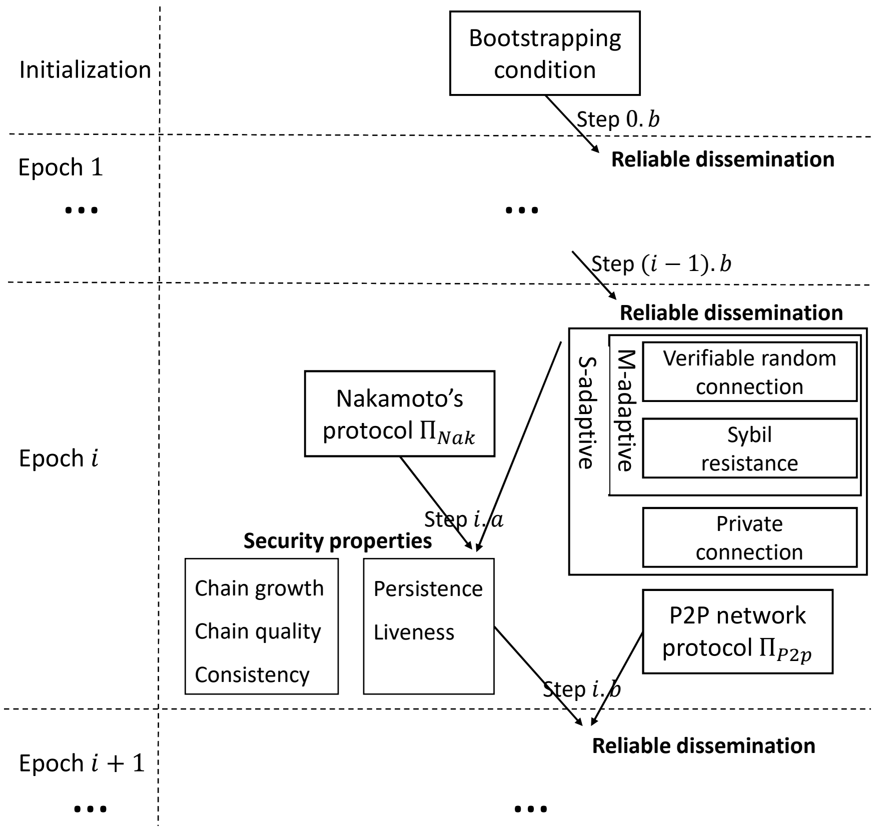


Fig. 4: The induction proof of the CoSpaN protocol.

Let $e_{\max}$ be the number of epochs in the execution. For $i \in [0..e_{\max}]$, let $\textsc{view}_i$ be the view of nodes in $\textsc{view}$ up until the last round of epoch i (the last round in which there exists an honest node with the chain of length $i \cdot L$). For $\Delta = O(\log n)\delta$, we define $\mathcal{S}_i, \mathcal{R}_i$ as the events where protocol Π_{CSN} achieves the *security properties* and *reliable dissemination*, respectively, up until epoch i:

$$\mathcal{S}_i \stackrel{\text{def}}{=} (\textsc{view} \leftarrow \mathsf{EXEC}_{\mathcal{A},\mathcal{Z}}^{\Pi_{\text{CSN}}}(\kappa) : \mathsf{sec}(\textsc{view}_i) = 1),$$
$$\mathcal{R}_i \stackrel{\text{def}}{=} (\textsc{view} \leftarrow \mathsf{EXEC}_{\mathcal{A},\mathcal{Z}}^{\Pi_{\text{CSN}}}(\kappa) : \mathsf{rel}(\textsc{view}_i, \Delta) = 1).$$

In the induction proof, for each epoch $j \in [1..e_{\max}]$, we prove

$$\Pr[\mathcal{S}_j \wedge \mathcal{R}_{j+1}] > 1 - (2j+1)\varepsilon(\kappa).$$

The statement for $j = e_{\max}$ indicates that protocol Π_{CSN} achieves the security properties.

Basis step (step 0-B). We assume a *bootstrapping condition* where the network of honest nodes is connected with a small diameter at epoch 1. Thus, protocol Π_{CSN} achieves reliable dissemination at epoch 1, i.e., $\Pr[\mathcal{R}_1] > 1 - \varepsilon(\kappa)$. The bootstrapping condition can be achieved through a trusted setup or a one-time decentralized bootstrapping scheme.

Induction step. Given that the hypothesis is true for $j = i - 1$, (where $i \in [1..e_{\max}]$), we will prove the statement is true of $j = i$. The induction step consists of two parts as follows.

– *Step i-A:* Given from step $(i-1)$-B that

$$\Pr[\mathcal{S}_{i-1} \wedge \mathcal{R}_i] > 1 - (2i-1)\varepsilon(\kappa).$$

We prove $\Pr[\mathcal{S}_i \wedge \mathcal{R}_i] > 1 - 2i\varepsilon(\kappa)$ by showing that

$$\Pr[\mathcal{S}_i \mid \mathcal{S}_{i-1} \wedge \mathcal{R}_i] > 1 - \epsilon(\kappa).$$

– *Step i-B:* Given from step i-A that

$$\Pr[\mathcal{S}_i \wedge \mathcal{R}_i] > 1 - 2i\varepsilon(\kappa).$$

We prove $\Pr[\mathcal{S}_i \wedge \mathcal{R}_{i+1}] > 1 - (2i+1)\varepsilon(\kappa)$ by showing that

$$\Pr[\mathcal{R}_{i+1} \mid \mathcal{S}_i \wedge \mathcal{R}_i] > 1 - \epsilon(\kappa).$$

Step *i*-A: Achieving the security properties from reliable dissemination Assuming that Π_{CSN} has achieved the security properties up until epoch $i-1$, we will now prove that it also achieves security properties up until epoch i. First, we summarize the security analysis of Nakamoto's protocol in the PSS model [26]. Then, we can reduce the security of Π_{CSN} in the SNM model to the security of Nakamoto's protocol Π_{Nak} in the PSS model, provided that the CoSpaN protocol achieves reliable dissemination.

Nakamoto's protocol in PSS model [26]. We summarize the security analysis of Nakamoto's protocol Π_{Nak} in PSS model [26], where nodes can send messages to all other nodes within a bounded time. We then demonstrate that the security of the CoSpaN protocol Π_{CSN} can be reduced to the security of Π_{Nak} if the reliable dissemination property is achieved. We say a tuple $(n, \rho, \Delta, \mathcal{A}, \mathcal{Z})$ with $\Gamma_{\text{PSS}}(n, \rho, \Delta) = 1$ is Γ_{PSS}-admissible w.r.t (Π, Λ) if reliable dissemination is given as an assumption, i.e., any message sent by an honest node is delayed by at most Δ rounds before sending to all other nodes. Further, Γ_{PSS}-admissible environments, in each round, each node can make 1 query to the random oracle (flat mining model).

Let $p = \frac{\mathsf{T}}{2^\kappa}$ be the probability that a node can generate a new block with a single query. We consider the following two quantities. Let $\alpha(n, \rho, \Delta) = 1 - (1-p)^{(1-\rho)n}$ be the probability that at least one honest node generates a new block in one round. Let $\beta(n, \rho, \Delta) = \rho n p$ be the expected number of blocks that the adversary can mine in a round. When the n, ρ, Δ is clear from the context, we simply write α, β. The analysis in [26] considers environments with a predicate Γ^*_{PSS} with the *honest majority assumption.* For $n, \Delta \in \mathbb{N}$, $\rho \in (0,1)$, we say $\Gamma^*_{\text{PSS}}(n, \rho, \Delta) = 1$ if $\alpha(1 - 2(\Delta+1)\alpha) > \beta$. We can achieve the security properties for ledger as follows.

Lemma 1 (Theorem 7.4 in [26]). *Consider protocol Π_{Nak} in Γ^*_{PSS}-admissible environments. Protocol Π_{Nak} maintains a ledger that satisfies persistence and liveness with the "wait time" $\mathtt{t} = \frac{\kappa}{(1-\epsilon)\alpha}$.*

In Pass et al. [26], the security properties for ledger are proven based on three *security properties for consensus* called *chain growth*, *consistency*, and *chain quality*. The formal definitions of those properties are presented in Appendix B.2.

CoSpaN in SNM model (given reliable dissemination). Given the event $\mathcal{R}_i$, which signifies that protocol Π_{CSN} achieves reliable dissemination up until epoch i, we can reduce the security of Π_{CSN} in the SNM model to that of Nakamoto's protocol Π_{Nak} in the PSS model. Intuitively, since any valid message can be disseminated within a bounded time Δ, we can simulate an execution in the PSS model to achieve the same message delay as in the SNM model. Additionally, in the non-flat mining model, each node v can be considered as a cluster of multiple nodes in the flat mining model. Similar to [26], we consider environments with a predicate Γ^*_{stat} that assumes an honest majority. For $n, Q, \delta \in \mathbb{N}$, $\eta, \rho \in (0, 1)$; we say $\Gamma^*_{\mathrm{stat}}(n, Q, \eta, \rho, \delta) = 1$ if there exists some $\Delta \in \mathbb{N}$ such that, $\Delta = O(\log n)\delta$ and $\Gamma^*_{\mathrm{PSS}}(Q, \rho, \Delta) = 1$.

Lemma 2 (Step *i*-A). *Consider protocol Π_{CSN} in Γ^*_{stat}-admissible environments. For any $i \in \mathbb{N}$, we have,* $\Pr[\mathcal{S}_i \mid \mathcal{S}_{i-1} \wedge \mathcal{R}_i] > 1 - \epsilon(\kappa)$.

The proof of Lemma 2 is presented in Appendix B.3.

Step *i*-B: Achieving reliable dissemination from the security properties Given that protocol Π_{CSN} achieves the security properties *up until epoch* i, we will prove that protocol Π_{CSN} achieves reliable dissemination *up until epoch* $i+1$. To ensure reliable dissemination, we demonstrate that the diameter of the network of honest nodes is bound by $O(\log n)$. This is achieved by showing that the number of *honest core nodes* is sufficiently large and the connections between honest nodes are *verifiably random*. We then bound the diameter of the network of honest nodes by proving 1) the network of honest core nodes has a connected diameter of $O(\log n)$, and 2) each periphery node is connected to at least one honest core node.

Honest core nodes. We demonstrate that the number of honest core nodes is sufficiently large. The selection of core nodes through a PoW mechanism implies that the number of honest core nodes is proportional to the total mining power of honest nodes. Therefore, a big enough core width can ensure that the number of honest core nodes is high enough. To be more specific, in each round, malicious nodes can query the random oracle at most ρQ times, while honest nodes can query it at least $(1-\rho)$ times. According to Eq. 2, the probability of successfully generating a core registration with a single query to the random oracle is $p' = \frac{\mathtt{T}_{\mathrm{core}}}{2^\kappa}$. Thus, the expected number of core registrations of honest nodes in each round is at least $(1-\rho)Q \cdot p'$. Also, a fraction ω of honest nodes are willing to participate as core nodes. Therefore, in each round, the expected number of core registrations generated by honest nodes willing to participate as core nodes is $(1-\rho)\omega \cdot Q \cdot p' = (1-\rho)\omega \cdot Q \cdot p\frac{s}{L_{\mathrm{reg}}}$ (as $p = \frac{\mathtt{T}}{2^\kappa}$ and $\mathtt{T}_{\mathrm{core}} = \frac{s}{L_{\mathrm{reg}}}\mathtt{T}$, we replace $p' = p\frac{s}{L_{\mathrm{reg}}}$).

Nodes can participate as core nodes in the next epoch if their core registrations are included in the registration period of $L_{\mathtt{reg}}$ blocks. From the chain quality property, there is at least one honest block in the last k blocks. Thus, all honest core registrations generated within the first $L_{\mathtt{reg}} - k$ blocks (before the last honest block is generated) will be included in the registration period. It takes nodes at least $(L_{\mathtt{reg}} - k)(1 - o(1))/(Q \cdot p)$ time to generate $L_{\mathtt{reg}} - k$ blocks. Therefore, in each epoch, the number of honest core nodes is at least $(L_{\mathtt{reg}} - k)(1 - o(1))/(Q \cdot p) \cdot (1-\rho)\omega \cdot Q \cdot p\frac{s}{L_{\mathrm{reg}}} = \mathsf{q} \cdot s$, where $\mathsf{q} = (1-o(1))(1-\rho)\omega(1-\frac{k}{L_{\mathrm{reg}}})$.

Lemma 3 (Honest core nodes). *Consider any epoch i, let h_{c} be the number of honest core nodes. We have,* $\Pr[h_{\mathrm{c}} \geq \mathsf{q} \cdot s \mid \mathcal{S}_i \wedge \mathcal{R}_i] > 1 - e^{-\Omega(\kappa)}$, *where* $\mathsf{q} = (1-\rho)\omega(1 - \frac{1}{L_{\mathrm{reg}}}) - \epsilon$, *and* $\epsilon > 0$.

The proof of Lemma 3 is presented in Appendix B.4.

Verifiable random connections. The verifiable random connections property states that the probability of there being a connection from an honest core node to another honest node (core or periphery) is the same. In CoSpaN protocol, a node core can establish a connection to another node if it can compute a VRF output of the randomness and the other node's identity that is smaller than a given threshold. Based on the consistency property of the blockchain, all nodes will obtain the same randomness value. Additionally, as the output of VRFs is random, we can ensure that the probability of there being a connection from an honest core node to another honest node is the same.

Lemma 4 (Verifiable random connections). *For any honest node $u \in \bar{\mathsf{H}}$ and any honest core node $v \in \bar{\mathsf{H}}_{\mathrm{c}}$, let $\bar{\mu} = \frac{d}{2s} - \varepsilon(\kappa)$, we have,*

$$\Pr[X_{v \to u} = 1 \mid \mathcal{S}_i \wedge \mathcal{R}_i] \geq \bar{\mu}. \tag{4}$$

The proof of Lemma 4 is presented in Appendix B.4.
Reliable dissemination. We prove that the Π_{CSN} protocol achieves reliable dissemination by demonstrating that the diameter of the network among honest nodes is bounded by $O(\log n)$. We establish this result in two steps. First, we model the network of honest core nodes as an Erdos-Renyi random network. Following Theorem 7.1 in [13], we can show that the diameter of the network of honest core nodes is bounded by $O(\log n)$. Secondly, each honest core node will connect to any periphery node with a certain probability. Thus, with a sufficient number of core nodes, each honest periphery node will be connected to at least one honest core node.

Lemma 5 (Step i-B). *Let $\Delta = O(\log n) \times \delta$, $k = \Omega(\kappa)$. Consider protocol Π_{CSN} with parameters $L > \frac{k}{(1-\rho)\omega} + 2k$, $s > k$ and $d \geq \max(\frac{k}{-\log \mathsf{q}}, 4 \log s)$. We have, $\Pr[\mathcal{R}_{i+1} \mid \mathcal{S}_i \wedge \mathcal{R}_i] > 1 - \epsilon(\kappa)$.*

The proof of Lemma 5 is presented in Appendix B.4.

4.2 Network sparsity

We analyze the sparsity of the CoSpaN network. In the CoSpaN protocol, nodes establish verifiable random connections to other nodes based on public randomness. If the randomness is fixed, we can use a concentration inequality to bound the number of connections in the network. However, an adversary can perform a *randomness grinding attack*, i.e., use different values of randomness to increase the chance of connecting malicious nodes to honest nodes and increasing the sparsity. Fortunately, based on the chain quality property, the adversary cannot try too many different values of randomness. Thus, the sparsity of the network remains bounded.

Lemma 6 (Network sparsity). *For any $\epsilon > 0$, with probability $1 - e^{\Omega(d)}$, the average degree (sparsity) of an honest node in G is at most $D = \frac{2(1+\epsilon)}{(1-\eta)}(1 + \frac{s}{n})d$.*

We present the proof of Lemma 6 in Appendix B.5.

4.3 Main theorem

We are now ready to prove the security of CoSpaN protocol in a sparse network by induction as follows. From the bootstrapping condition and Lemma 2, we have protocol Π_{CSN} is secure in the bootstrapping epoch. Then, combining Lemma 2 and Lemma 5, if protocol Π_{CSN} is secure in epoch $i-1$, it is secure in epoch i.

Theorem 1 (Static security in a sparse network). *Consider $\Gamma^{*}_{\mathrm{stat}}$-admissible environments and protocol Π_{CSN} with parameters $L > \frac{k}{(1-\rho)\omega} + 2k$, $s > k$ and $d \geq \max(\frac{k}{-\log \mathsf{q}}, 4 \log s)$. Under the bootstrapping condition is satisfied, protocol Π_{CSN} achieves*

- *persistence, liveness ; and*
- *D-sparsity, where $D = 2(1+\epsilon)(1+\frac{s}{n})/(1-\eta)d$.*

The proof of Theorem 1 is presented in Appendix B.6.

5 Weakly Adaptive security

In this section, we demonstrate the security of the CoSpaN protocol in several (weakly) adaptive settings. In Subsection 5.1, we consider an *M-adaptive adversary* that requires some time to corrupt honest nodes. Then, in Subsection 5.2, we examine an *S-adaptive adversary* that can instantly corrupt honest nodes. In Appendix 5.3, we conduct further analysis on the security of the protocol in the presence of an S-adaptive adversary, particularly when the mining power is concentrated on a few top nodes.

5.1 M-adaptive security

We extend the model in Section 2 in the presence of a *mildly adaptive* (M-adaptive) adversary that can corrupt honest nodes during the protocol execution in which the time it takes to corrupt honest nodes is τ_{corr} rounds. We can prove that protocol Π_{CSN} achieves the same security properties as in the static setting (Section 4).

Protocol Π_{CSN} can defend against an M-adaptive adversary due to the fact that *the network topology dynamically changes* in every epoch. According to the chain growth property, within τ_{corr} rounds, the length of the chain of honest players increases by at least $2L$ blocks. In other words, it takes at least 2 epochs for the adversary to corrupt honest nodes. In order to corrupt some honest nodes at epoch i, the adversary must begin the corruption before epoch $i-2$. Since the core nodes of epoch i are selected at epoch $i-1$, the adversary *cannot predict the network topology* of epoch i at epoch $i-2$. Therefore, the adversary cannot gain an advantage by adaptively corrupting the honest nodes.

We consider admissible environments in the presence of an M-adaptive adversary (see Appendix C.1 for formal definition) with a predicate $\Gamma^*_{\text{M-adap}}$ in which the honest majority assumption is given and $\tau_{\text{corr}} > 2\left(\frac{k}{(1-\rho)\omega} + 2k\right) / \left((1-\epsilon)\,\alpha\right)$.

Theorem 2 (M-adaptive security). *Let $k = \Omega(\kappa)$. Consider $\Gamma^*_{\text{M-adap}}$-admissible environments and protocol Π_{CSN} with parameters $\frac{k}{(1-\rho)\omega} + 2k < L < \frac{\tau_{\text{corr}}(1-\epsilon)\alpha}{2}$, $\epsilon \in (0,1)$, $s > k$ and $d \geq \max(\frac{k}{-\log \mathsf{q}}, 4\log s)$. Under the bootstrapping condition, protocol Π_{CSN} achieves persistence and liveness; and sparsity.*

The proof of Theorem 2 is presented in Appendix C.1.

5.2 S-adaptive security

We analyze the security of the CoSpaN protocol against a *slow-observation adaptive* (S-adaptive) adversary that can instantly corrupt honest nodes, but takes τ_{obs} rounds to observe connections among honest nodes. The S-adaptive adversary is stronger than the M-adaptive adversary when $\tau_{\text{obs}} = \tau_{\text{corr}}$. The M-adaptive adversary needs to start corrupting some honest nodes at round $r - \tau_{\text{corr}}$ to corrupt them at round r, while the S-adaptive adversary can start at round r. Therefore, the S-adaptive adversary can gather more information to decide which nodes to corrupt. To defend against an S-adaptive adversary, we need to increase the core width, which in turn increases the network sparsity.

We consider admissible environments in the presence of S-adaptive adversaries with a predicate $\Gamma^*_{\text{S-adap}}$ in which the honest majority assumption is given and $\tau_{\text{obs}} > \frac{4k}{\alpha} + \frac{2k}{(1-2\rho)\omega\cdot\alpha}$.

We set the core width to ensure that most honest nodes are selected as core nodes. This way, even if the adversary can corrupt a set of nodes that control a fraction ρ of mining power, we can still guarantee a large enough number of honest core nodes.

Lemma 7 (Honest core nodes in S-adaptive security). *For any $\epsilon > 0$, let $\mathsf{q}_{\text{M-adap}} = (1-2\rho)\omega - \frac{k}{L_{\text{reg}}} - \epsilon$. Consider protocol Π_{CSN} with parameter $s > k + \frac{2(1+\epsilon)}{\mathsf{q}_{\text{M-adap}}\cdot\epsilon^2 \log_2 e} h$, we have,*

$$\Pr[h_{\text{c}} \geq s \cdot \mathsf{q}_{\text{M-adap}} \mid \mathcal{S}_i \wedge \mathcal{R}_i] > 1 - e^{-\Omega(\kappa)}.$$

The proof of Lemma 7 is presented in Appendix C.2.

Next, we demonstrate that the connections between honest core nodes and other honest nodes are verifiably random and unpredictable. This means the adversary has no knowledge of whether there are connections among honest nodes.

Lemma 8 (Private connection in S-adaptive security). *Let $\bar{\mathsf{H}}$ be the set of honest (core and periphery) nodes and $\bar{\mathsf{H}}_c \subseteq \bar{\mathsf{H}}$ be the set of honest core nodes at the beginning of epoch i. Until some round r at epoch i, let $\bar{\mathsf{W}} \subseteq \bar{\mathsf{H}}$ be the set of nodes that is corrupted by the adversary. Let $\bar{\mathsf{W}}_c \subseteq \bar{\mathsf{W}}$ be the set of corrupted core nodes. Consider an adversary $\mathcal{A}$ runs some* PPT *algorithm to return an honest node $u \in \bar{\mathsf{H}} \setminus \bar{\mathsf{W}}$ and an honest core node $v \in \bar{\mathsf{H}}_c \setminus \bar{\mathsf{W}}_c$. We have,*

$$\Pr[X_{v \to u} = 1 \mid \mathcal{S}_i \wedge \mathcal{R}_i] \leq \bar{\mu} + \varepsilon(\kappa). \tag{5}$$

Private connections ensure that an adversary cannot take advantage by adaptively corrupting honest nodes. This ensures that connections remain verifiably random, even in the presence of an S-adaptive adversary.

Lemma 9 (Random verifiable connection in S-adaptive security). *For any honest node $u \in \bar{\mathsf{H}} \setminus \bar{\mathsf{W}}$ and any honest core node $v \in \bar{\mathsf{H}}_c \setminus \bar{\mathsf{W}}_c$, we have,*

$$\Pr[X_{v \to u} = 1 \mid \mathcal{S}_i \wedge \mathcal{R}_i] \geq \bar{\mu}. \tag{6}$$

The proofs of Lemmas 9 and 8 are in Appendix C.2.

Now, we can follow the same proof in the Lemma 5 (except replacing the core quality q by $\mathsf{q}_{\text{M-adap}}$) to prove that protocol Π_{CSN} achieves Δ-reliable dissemination. Then, following the same proofs and Theorem 1, we can prove the security of protocol Π_{CSN} in the presence of an S-adaptive adversary.

Theorem 3 (S-adaptive security). *Consider $\Gamma^*_{\text{S-adap}}$-admissible environments and protocol Π_{CSN} with parameter $s > k + \frac{2(1+\epsilon)}{\mathsf{q}_{\text{M-adap}} \cdot \epsilon^2 \log_2 e} h$, $\frac{k}{(1-2\rho)\omega} + 2k < L < \frac{\tau_{\text{obs}}(1-\epsilon)\alpha}{2}$, and $d \geq \max(\frac{k}{-\log \mathsf{q}_{\text{M-adap}}}, 4 \log s)$. Given that the bootstrapping condition is satisfied. Then, protocol Π_{CSN} achieves persistence and liveness; and sparsity.*

5.3 S-adaptive $\langle \phi, \gamma \rangle$ security

We consider *mining power concentration*[7], in which the top γ fraction of honest nodes control at least a fraction γ of the mining power, for some $\gamma > \rho$. With mining power concentrated in the top few nodes, it is possible to reduce the number of core nodes, thus reducing network sparsity. More concretely, since mining power is concentrated in a small fraction of nodes, the CoSpaN protocol requires a smaller core width so that the nodes selected as core control a large fraction of the mining power. The adversary can not control more than a fraction ρ of mining powers, we can guarantee that a big enough number of core nodes are honest.

We consider admissible environments with a predicate Γ^*_{conc} such that: for $n, Q, \delta, \tau_{\text{obs}} \in \mathbb{N}$, $\eta, \rho, \phi, \gamma \in (0,1)$, $\Gamma^*_{\text{conc}}(n, Q, \eta, \rho, \delta, \tau_{\text{obs}}, \phi, \gamma) = 1$ if $\Gamma^*_{\text{S-adap}}(n, Q, \eta, \rho, \delta, \tau_{\text{obs}}) = 1$ and $\gamma > \rho$.

Theorem 4 (S-adaptive $\langle \phi, \gamma \rangle$). *Consider Γ^*_{conc}-admissible environments and the protocol Π_{CSN} with parameters $\frac{k}{(\gamma-\rho)\omega} + 2k < L < \frac{\tau_{\text{obs}}(1-\epsilon)\alpha}{2}$, $s > k + h\phi$ and $d \geq \max(\frac{k}{-\log \mathsf{q}_{\text{conc}}}, 4 \log s)$. Under the bootstrapping condition is satisfied, protocol Π_{CSN} achieves persistence and liveness; and sparsity.*

The proof of theorem 4 is presented in Appendix C.3.

6 Numerical Studies

We compare the CoSpaN protocol to existing Bitcoin-like protocols by analyzing the costs for adversaries to perform various attacks. Additionally, we analyze the defense costs in the proposed threat models and the effect of mining power concentration on reducing defense costs.

[7] For example, in Bitcoin [1], more than 50% of the mining power is controlled by the top 7 mining pools.

6.1 Setup

Protocols. We consider CoSpaN and the following two protocols.

- Bitcoin-L(ike):This protocol mimics the current protocol for the Bitcoin P2P network. Each node makes $d = 8$ outbound connections and accepts up to $d_{max} = 125 \approx 15d$ inbound connections. Nodes follow the proposed defenses in [17] to randomize their connections, i.e., 1) each node makes connections to $d = 8$ random nodes, and 2) if a node receives more than d_{max} requests for inbound connections, it randomly selects d_{max} nodes to accept the connections. Note that the adversary can no longer perform preferential attachment attacks to increase the probability of honest nodes establishing outbound connections toward their controlled nodes. However, we assume that each adversary-controlled node will make connections to *all* honest nodes to get more slots from the 125 inbound connections to honest nodes.
- Bitcoin-R(andom): This protocol considers a more advanced topology construction, building a truly random topology over the participating nodes. Any two nodes have exactly the same probability $\frac{d}{n}$ of having a connection. Thus, the expected number of connections for each node is d, and there is no limit on the number of inbound connections. In this protocol, the adversary can no longer get more slots within the 125 connection limit of the honest nodes. However, the adversary can increase the number of connections to honest nodes by creating more sybil nodes (higher η).

Note with the same connection parameter d, all three protocols will construct networks of similar sparsity (average degree).

Parameter settings. We consider a network with $1,000$ honest nodes, $1,000$ malicious nodes, and additional $2,000 \cdot n_s$ Sybil nodes (of zero-mining power). Thus, the fraction of malicious node $\eta = \frac{1}{2(1+n_s)}$. Without otherwise mentioned, we consider a static adversary that controls a fraction $\rho = 0.3$ of mining power, the participation rate in CoSpaN protocol is $\omega = 0.9$. For CoSpaN protocol, we set the core width to be 5% the number of nodes in the network.

Metrics. We evaluate the adversary's cost to break protocol security via: (1) the *Sybil factor* required to disconnect the network (violating reliable dissemination), and (2) the *adversarial mining power* needed for a double spending attack. We also analyze the *honest mining power* required for CoSpaN security under various settings, along with the *degree distribution* and its correlation with mining power. All experiments were repeated 1,000 times, and averages are reported.

6.2 Costs of attacks

We compare the protocols by showing the cost for the adversary to perform certain attacks in static settings.

Sybil attacks We consider a static adversary that controls 30% of the mining power. The adversary attempts to disrupt the network by creating more Sybil nodes, i.e., increasing the Sybil factor, in order to break the reliable dissemination. As shown in Fig. 5, the cost to break the reliable dissemination of the CoSpaN protocol is significantly higher than that of the other protocols. For example, at Bitcoin's sparsity level with $d = 8$, the adversary needs a Sybil factor $n_s > 100$ to break the reliable dissemination of the CoSpaN protocol, while the numbers for Bitcoin-L and Bitcoin-R are only $n_s = 0$ and $n_s = 1.5$, respectively. Even at a much higher connection parameter $d = 64$, the adversary can still break the reliable dissemination security requirement of Bitcoin-L and Bitcoin-R with a high Sybil factor ($n_s < 100$).

Double-spending attacks We consider a static adversary with a Sybil factor of $n_s = 10$ and a connection parameter of $d = 8$. The adversary performs a double-spending attack by 1) splitting the network, and thus the honest mining power, and 2) using the block withholding strategy [26]. In Fig. 6, we show the required mining power for the adversary to successfully reverse a 12-confirmation transaction with a probability of more than 10%. For the CoSpaN protocol, the adversary always needs 51% of the mining power to perform the attacks, as it cannot split the network. For the Bitcoin-L and Bitcoin-R protocols, the honest nodes are disrupted into multiple components. Hence, the adversary can perform the double-spending attack in each component with much less mining power. For example, if the Sybil factor is $n_s = 100$ and the connection

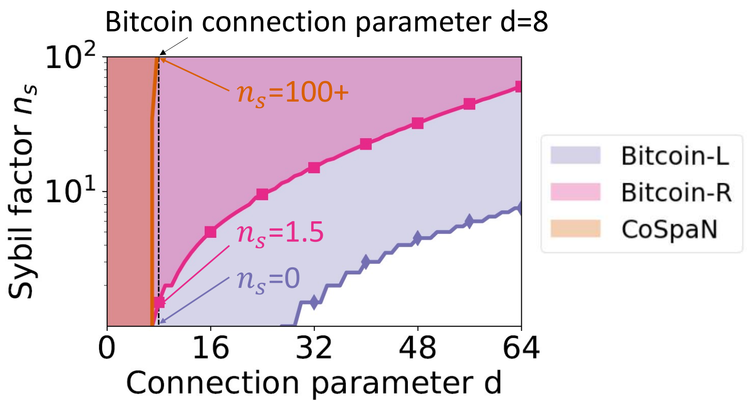


Fig. 5: Protocols' resiliency against Sybil attacks. For each network connection parameter (d), the shaded areas above the curves show the corresponding Sybil factors that the adversary needs to break the protocols' security. The higher Sybil factor that a protocol can withstand, the stronger security. At the Bitcoin's connection parameter $d = 8$ (the dashed line), the adversary can disrupt Bitcoin-L's network without any Sybil nodes ($n_s = 0$), disrupt Bitcoin-R's network with $n_s \approx 1.5$, but can only disrupt CoSpaN's network for unrealistically high $n_s > 100$.

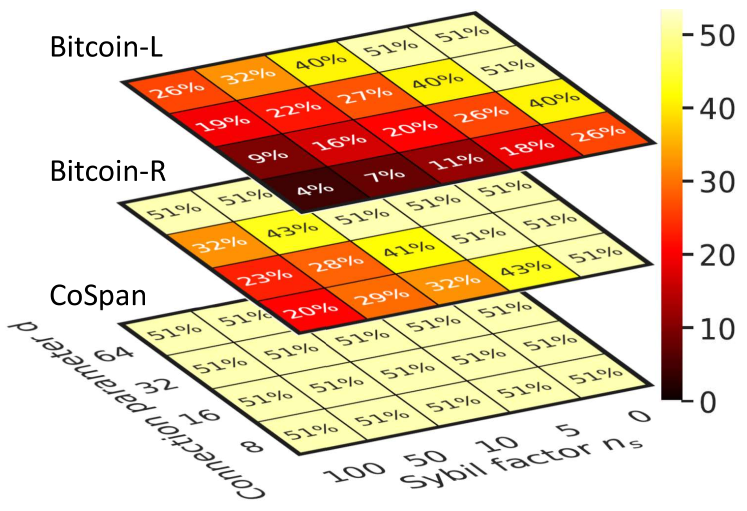


Fig. 6: Double-spending attacks with less than 51%. The required mining power by an adversary to reverse a 12-confirmation transaction with a more than 10% probability in each protocol. The adversary is able to split the networks in Bitcoin-L and Bitcoin-R (but not CoSpaN) protocols, thus, only needs to exceed the mining power of the largest connected component.

parameter is $d = 8$, the adversary only needs 4% of the mining power to attack the Bitcoin-L network and 20% of the mining power to attack the Bitcoin-R network.

6.3 CoSpaN in different security settings

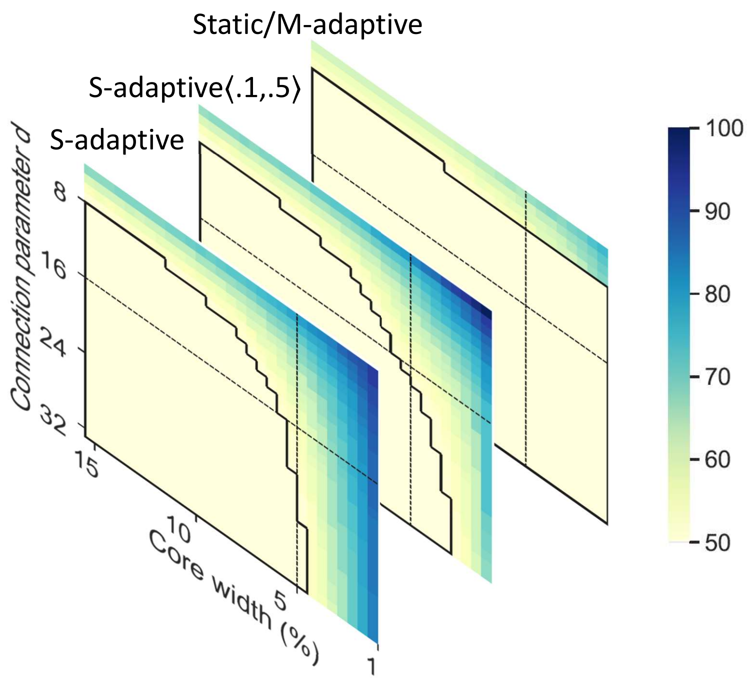


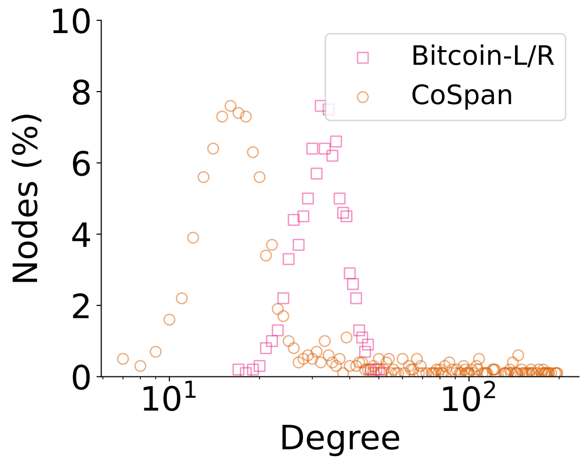
Fig. 7: The required honest mining power to achieve reliable dissemination for CoSpaN protocol. The area under the curve show when the same requirement (51%) in PSS is met.

We will now analyze the security of the CoSpaN protocol in different settings. We will omit Bitcoin-L and Bitcoin-R as they cannot achieve the desired security properties in (weakly) adaptive settings. Figure 7 shows the required honest mining power to achieve security in the network layer of the CoSpaN protocol with different parameter settings (connection parameter and core width). We will consider four security settings: static, M-adaptive, S-adaptive (with uniform mining power), and S-adaptive $\langle 0.1, 0.5\rangle$.

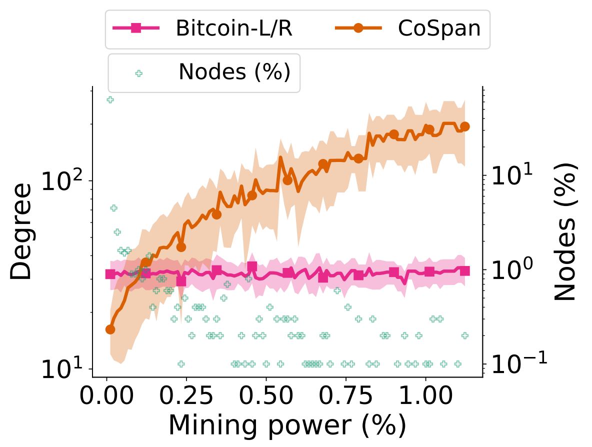
In the S-adaptive setting, there is a trade-off between the connection parameter and the core width. When the core width increases, it requires a smaller connection parameter, and vice versa. For example, when the core width increases from 5% to 15% to achieve the security of 51% of honest mining power, the connection parameter d decreases from 22 to 8. Mining power concentration helps to reduce the required core width. For example, with a core width of 5%, the required connection parameter in an S-adaptive setting with no mining power concentration (i.e., all nodes have the same mining power) is 22. Meanwhile, the required connection parameter in S-adaptive $\langle 0.1, 0.5\rangle$ is 17.

6.4 Network characteristics

Consider a network in which the mining power of nodes follows a power law distribution. We analyze the degree distribution and the relation between the degree distribution and the mining power distribution of the honest nodes. From Fig. 8a, the degree distribution of Bitcoin-L and Bitcoin-R networks follows a normal distribution. The degree distribution of the CoSpaN network follows a power law distribution. Specifically, as shown in Fig. 8, the degrees of nodes in Bitcoin-L and Bitcoin-R networks are independent of the mining power. The degree of nodes in those networks has the same distribution. Meanwhile, in the CoSpaN network, the degree of nodes is proportional to the mining power.



(a) Degree distribution distribution.



(b) The correlation between mining power and degree of nodes. The shaded areas show the 99% confidence intervals.

Fig. 8: The correlation between the mining power and the degree of nodes in CoSpaN and Bitcoin-random network. The distribution of Bitcoin-L and Bitcoin-R are the same.

7 Conclusion

This paper presents CoSpaN, the first network protocol designed for sparse networks in the PoW setting. Nodes use merge-mining to provide proofs of their mining power, and establish verifiable random connections such that the expected number of connections for each node is proportional to its mining power. Additionally, the CoSpaN protocol preserves the confidentiality of core nodes, meaning that the adversary cannot determine whether or not a node is associated with a core node. As a result, the adversary is unable to launch DoS attacks to bring down core nodes and disrupt the network. Plus, each connection is known only by the two participating nodes. This is a key feature for achieving reliable dissemination in a sparse network against an adaptive adversary.

Our CoSpaN protocol can be utilized to build a backbone network for the dissemination of critical information such as block headers. Additional connections can be added to increase the throughput and performance of the blockchain systems. A future direction is to investigate whether or not we can construct a secure consensus protocol with nodes of bounded degree.

References

1. https://www.blockchain.com/pools, accessed 11/27/2021.
2. Vivek Kumar Bagaria, Sreeram Kannan, David Tse, Giulia C. Fanti, and Pramod Viswanath. Prism: Deconstructing the blockchain to approach physical limits. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 585–602. ACM Press, November 2019.
3. Soumya Basu, Ittay Eyal, and Emin Sirer. Falcon. https://www.falcon-net.org/.
4. Alex Biryukov, Dmitry Khovratovich, and Ivan Pustogarov. Deanonymisation of clients in bitcoin p2p network. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pages 15–29, 2014.
5. Joseph Bonneau, Andrew Miller, Jeremy Clark, Arvind Narayanan, Joshua A. Kroll, and Edward W. Felten. SoK: Research perspectives and challenges for bitcoin and cryptocurrencies. In *2015 IEEE Symposium on Security and Privacy*, pages 104–121. IEEE Computer Society Press, May 2015.
6. Matt Corallo. Bitcoin relay network.
7. Sandro Coretti, Aggelos Kiayias, Cristopher Moore, and Alexander Russell. The generals scuttlebutt: Byzantine-resilient gossip protocols. *CCS*, 2022:541, 2022.

8. Bernardo David, Peter Gazi, Aggelos Kiayias, and Alexander Russell. Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 66–98. Springer, Heidelberg, April / May 2018.
9. Christian Decker and Roger Wattenhofer. Information propagation in the bitcoin network. In *Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on*, pages 1–10. IEEE, 2013.
10. John Dillon. Bitcoin-development mailinglist: Protecting bitcoin against network-wide dos attack, 2018.
11. Yevgeniy Dodis and Aleksandr Yampolskiy. A verifiable random function with short proofs and keys. In Serge Vaudenay, editor, *PKC 2005*, volume 3386 of *LNCS*, pages 416–431. Springer, Heidelberg, January 2005.
12. Danny Dolev. The byzantine generals strike again. *Journal of algorithms*, 3(1):14–30, 1982.
13. Alan Frieze and Michał Karoński. *Introduction to random graphs*. Cambridge University Press, 2016.
14. Juan A. Garay, Aggelos Kiayias, and Nikos Leonardos. The bitcoin backbone protocol: Analysis and applications. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 281–310. Springer, Heidelberg, April 2015.
15. Arthur Gervais, Ghassan O Karame, Karl Wüst, Vasileios Glykantzis, Hubert Ritzdorf, and Srdjan Capkun. On the security and performance of proof of work blockchains. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 3–16, 2016.
16. Michel Goemans. Chernoff bounds, and some applications. *https://math.mit.edu/~goemans/18310S15/chernoff-notes.pdf*, 2015.
17. Ethan Heilman, Alison Kendler, Aviv Zohar, and Sharon Goldberg. Eclipse attacks on bitcoin's peer-to-peer network. In Jaeyeon Jung and Thorsten Holz, editors, *USENIX Security 2015*, pages 129–144. USENIX Association, August 2015.
18. Valerie King and Jared Saia. From almost everywhere to everywhere: Byzantine agreement with õ(n3/2) bits. In *International Symposium on Distributed Computing*, pages 464–478. Springer, 2009.
19. Chen-Da Liu-Zhang, Christian Matt, Ueli Maurer, Guilherme Rito, and Søren Eller Thomsen. Practical provably secure flooding for blockchains. In *Advances in Cryptology–ASIACRYPT 2022: 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5–9, 2022, Proceedings, Part I*, pages 774–805. Springer, 2023.
20. Chen-Da Liu-Zhang, Christian Matt, and Søren Eller Thomsen. Asymptotically optimal message dissemination with applications to blockchains. *Cryptology ePrint Archive*, 2022.
21. Yuval Marcus, Ethan Heilman, and Sharon Goldberg. Low-resource eclipse attacks on ethereum's peer-to-peer network. *IACR Cryptol. ePrint Arch.*, 2018:236, 2018.
22. Christian Matt, Jesper Buus Nielsen, and Søren Eller Thomsen. Formalizing delayed adaptive corruptions and the security of flooding networks. In *Advances in Cryptology–CRYPTO 2022: 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15–18, 2022, Proceedings, Part II*, pages 400–430. Springer, 2022.
23. Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008. https://bitcoin.org/bitcoin.pdf.
24. Gleb Naumenko, Gregory Maxwell, Pieter Wuille, Alexandra Fedorova, and Ivan Beschastnikh. Erlay: Efficient transaction relay for bitcoin. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 817–831. ACM Press, November 2019.
25. Kartik Nayak, Srijan Kumar, Andrew Miller, and Elaine Shi. Stubborn mining: Generalizing selfish mining and combining with an eclipse attack. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 305–320. IEEE, 2016.
26. Rafael Pass, Lior Seeman, and abhi shelat. Analysis of the blockchain protocol in asynchronous networks. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 643–673. Springer, Heidelberg, April / May 2017.
27. Rafael Pass and Elaine Shi. Hybrid consensus: Efficient consensus in the permissionless model. In *31st International Symposium on Distributed Computing (DISC 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.
28. Peter R Rizun. Subchains: A technique to scale bitcoin and improve the user experience. *Ledger*, 1:38–52, 2016.
29. Muhammad Saad and David Mohaisen. Three birds with one stone: Efficient partitioning attacks on interdependent cryptocurrency networks. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 1404–1418. IEEE Computer Society, 2022.
30. Ye Wang and Roger Wattenhofer. Asynchronous byzantine agreement in incomplete networks. In *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*, pages 178–188, 2020.
31. Karl Wüst and Arthur Gervais. Ethereum eclipse attacks. Technical report, ETH Zurich, 2016.

A Supplemental for Section 3

A.1 Nakamoto's protocol Π_{Nak}

In Nakamoto's protocol Π_{Nak} [27], each node maintains a chain, i.e., a sequence of blocks linked by hash values. Each block is a tuple $(\mathsf{context}, txs, nonce)$, where txs the list of transactions that the node received from the environment, the $\mathsf{context}$ consists of a pointer (hash value) prev to the previous block, the Merkle root $MkRoot(txs)$, a single hash value to prove the validate the included transactions following the approach in the Bitcoin's header.

Also, we add to the context a *network pseudo-identity* cid, that is needed in core selection,

$$\mathsf{context} = \langle \mathsf{prev}, MkRoot(txs), \mathsf{cid} \rangle. \tag{7}$$

The algorithm Λ obtains the ledger from a chain $\mathcal{C}$ by truncating the last k blocks for some integer $k = O(\kappa)$ and returns the list of transactions [26]. The nodes compete to become *block producers* via a PoW mechanism. The block producer will generate and propagate a new block to all the nodes.

Block producer selection. In each round, nodes makes attempts to solve a PoW puzzle by searching for a *nonce* over a $\mathsf{context}$ that satisfies the hash inequality

$$H(\mathsf{context}, nonce) < \mathtt{T},$$

where $\mathtt{T}$ is the *mining difficulty.*

If the hash inequality is satisfied, the node becomes a block producer, adds a new block $(\mathsf{context}, txs, nonce)$ to the longest (best) chain, and propagate the new chain to all the neighbors (who will forward the chain further). Before adding a new chain $\mathcal{C}$ to the local state, each node verifies the validity of 1) the hash inequalities in all blocks and 2) the list of transactions in the chain, using a predicate V.

A.2 Verifiable Random Functions

In our design, a verifiable random function (VRF) has been used for core miners to establish connections with other miners. Here, we present formal definitions of Verifiable Random Functions in [11].

Definition 7. *A function family $F_{(\cdot)}(\cdot) : \{0,1\}^{\mathsf{a}(\kappa)} \to \{0,1\}^{\mathsf{b}(\kappa)}$ (where $\mathsf{a}(\cdot), \mathsf{b}(\cdot)$ are polynomials) is a family of VRFs if there exist algorithms (Gen, Prove, Verify) such that:*

- *The algorithm Gen takes as input a security parameter 1^κ and outputs a key pair $(\mathsf{sk}', \mathsf{pk}')$.*
- *The algorithm Prove takes as input a private key sk', a string x, and outputs a pair (σ, π), where $\sigma = F_{\mathsf{sk}'}(x)$.*
- *The algorithm Verify takes as input a public key pk', a string x, an output σ, a proof π and verifies that $\sigma = F_{\mathsf{sk}'}(x)$ using the proof π. It output 1 if y is valid and 0 otherwise.*

Additionally, we require the following properties:

1. ***Uniqueness.*** *No values $(\mathsf{pk}', x, \sigma, \sigma', \pi, \pi')$ can satisfy both*

$$\mathsf{Verify}_{\mathsf{pk}'}(x, \sigma, \pi) = 1 \text{ and } \mathsf{Verify}_{\mathsf{pk}'}(x, \sigma', \pi') = 1$$

   unless $\sigma = \sigma'$
2. ***Provability.*** *If $(\sigma, \pi) := \mathsf{Prove}_{\mathsf{sk}'}(x)$, then $\mathsf{Verify}_{\mathsf{pk}'}(x, \sigma, \pi) = 1$.*
3. ***Pseudorandomness.*** *For all PPT adversary $\mathcal{A} = (\mathcal{A}_E, \mathcal{A}_J)$ that does not query the oracle on x, we have*

$$\Pr\left[\begin{array}{l|l} (\mathsf{pk}', \mathsf{sk}') \leftarrow \mathsf{Gen}(1^\lambda); & \\ (x, st) \leftarrow \mathcal{A}_E^{\mathsf{Prove}_{\mathsf{sk}'}(\cdot)}; & \\ y_0 := F_{\mathsf{sk}'}(x); y_1 \leftarrow \{0,1\}^{\mathsf{b}(\kappa)}; & b = b' \\ b \leftarrow \{0,1\}; & \\ b' \leftarrow \mathcal{A}_J^{\mathsf{Prove}_{\mathsf{sk}'}(\cdot)}(y_b, st) & \end{array}\right] \leq \frac{1}{2} + \varepsilon(\lambda).$$

A.3 Discussions

We now discuss the practicality of CoSpaN protocol in real-world blockchain systems.

CoSpaN as a backbone network. We can use the CoSpaN protocol to construct a secure backbone network for propagating important information, such as block headers. For actual data propagation, a high-performance relay network such as FIBRE [6] or Falcon [3] can be used.

Coping with low participation rate of core nodes. In practice, some core nodes may not participate in establishing connections in the next epoch. This can be due to nodes being offline or behind a NAT, making them unreachable by other nodes. To address this issue without increasing the number of connections, we can prioritize the use of VRF outputs to establish or accept connections instead of relying on the inequalities. Specifically, nodes will establish or accept connections with the top few nodes that have the smallest VRF outputs. For example, suppose a node needs to connect with some core nodes but they did not participate in establishing connections. In that case, the node will instead connect with the core nodes that have the next smallest VRF outputs.

Public-private nodes. In Bitcoin network, nodes can be categorized into two types: public and private nodes. We can also view the Bitcoin network as a core-periphery network where the public nodes can be viewed as core nodes and the private nodes can be viewed as periphery nodes. This is contradict with the CoSpaN network where the core nodes are private and the periphery nodes are public.

The CoSpaN network is more resistant to DoS attacks, compared with the Bitcoin network. Indeed, in Bitcoin network, the IP addresses of core nodes are public (known by everyone); while the IP addresses of core nodes in CoSpaN network are private (only known by the nodes that have direct connections to them). Without knowing the IP address of core nodes, the adversary cannot perform DOS attacks to bring down those core nodes.

B Supplemental for Section 4

B.1 Chernoff bound

We provide here the Chernoff bound for Bernoulli random variables [16] that we use in the proofs.

Lemma 10. *Suppose $\{Z_i : i \in [t]\}$ are independent and identically distributed Bernoulli random variables with $\Pr[Z_i = 1] = \mu$, for all $i \in [t]$. Then, for any $\epsilon > 0$, we have*

$$\Pr[\sum_{i=1}^{t} Z_i \le (1-\epsilon)t\mu] \le e^{-\epsilon^2 t\mu/2},$$
$$\Pr[\sum_{i=1}^{t} Z_i \ge (1+\epsilon)t\mu] \le e^{-\epsilon^2 t\mu/3}.$$

B.2 Security analysis for Nakamoto's protocol in PSS model [26]

The PSS model We now describe the PSS model for a blockchain protocol execution. There are three main differences between the PSS model and the SNM model. First, the PSS model considers broadcast communication in which nodes can send messages to all other nodes. Secondly, in PSS model, all nodes have the same mining power, i.e., flat mining model. Finally, the PSS model consider a strong adversary that can adaptively corrupt honest nodes and obverse the messages between any nodes.

Participating nodes. The set of participating nodes N can be partitioned into the set of *honest nodes* $\mathsf{H} \subset \mathsf{N}$, who strictly follow the blockchain protocol (Π, Λ) and the set of *malicious nodes* $\mathsf{N} \setminus \mathsf{H}$, controlled by an *adversary* $\mathcal{A}$. The number of malicious nodes $f = |\mathsf{N} \setminus \mathsf{H}|$ is bounded by $\rho \cdot n$ for a fixed $\rho \in (0, 1/2)$.

A blockchain protocol. Algorithms Π and Λ have the same description as in SNM model in Section 2.

Flat mining model. All nodes interact with a random oracle $H : \{0,1\}^* \to \{0,1\}^\kappa$ that takes an arbitrary length string and output a random value in $\{0,1\}^\kappa$. In each round, a node $i \in \mathsf{N}$ can make one query to the random oracle H.

Broadcast communication. Nodes can send message to all other nodes in the network. The adversary $\mathcal{A}$ is responsible for delivering all messages. The adversary cannot forge or alter the messages of honest nodes. But it can delay or reorder the messages as long as the messages are delivered within some *unknown delivery delay* $\Delta \in \mathbb{N}$.

A blockchain execution. The blockchain execution of PSS model is the same as in SNM model except that the adversary can instantly corrupt honest nodes during the protocol execution.

Admissible environments. We consider executions with restricted adversaries and environments as follows.

Definition 8 (Γ_{PSS}-admissible environments). *We say a tuple $(n, \rho, \Delta, \mathcal{A}, \mathcal{Z})$ with $\Gamma_{\mathrm{PSS}}(n, \rho, \Delta) = 1$ is Γ_{PSS}-admissible w.r.t (Π, Λ) if $\mathcal{A}$, $\mathcal{Z}$ are probabilistic polynomial-time algorithms, and for every $\kappa \in \mathbb{N}$, for every view in the support of $\mathsf{EXEC}_{\Pi,\mathcal{A},\mathcal{Z}}(\kappa)$, the following holds:*

1. *The adversary $\mathcal{A}$ can control at most $\rho \cdot n$ nodes, where n is the total number of nodes, and each node has 1 mining power.*
2. *Conditions (2) in Def. 1.*
3. *Any message sent by an honest node is delayed by at most Δ rounds before sending to all other nodes.*

Security properties We present the definitions of chain growth, consistency, and chain quality in [26] as follows.

Chain growth. For a chain growth parameter cg and $\textsc{view} \leftarrow \mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa)$, we define $\mathsf{grw}(\textsc{view}, \mathsf{cg}) = 1$ iff for any honest player i at any rounds r and $r' = r + t$ (where $t = \Omega(\kappa)$), we have, $\mathsf{len}(\mathcal{C}_i^{r'}) - \mathsf{len}(\mathcal{C}_i^r) \geq t\mathsf{cg}$ where, $\mathcal{C}_i^r, \mathcal{C}_i^{r'}$ are the chains of the player i at round r, r', respectively.

Definition 9 (Chain growth). *A blockchain protocol $(\Pi^{\mathsf{V}}, \Lambda)$ achieves* chain growth *in Γ_{stat}*-environments *if there exists a chain growth parameter $\mathsf{cg} > 0$ such that*

$$\Pr[\textsc{view} \leftarrow \mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa) : \mathsf{grw}(\textsc{view}, \mathsf{cg}) = 1] \geq 1 - \varepsilon(\kappa),$$

in all Γ_{stat}-admissible environments.

Chain quality. For a chain quality parameter cq and $\textsc{view} \leftarrow \mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa)$, we define $\mathsf{qty}(\textsc{view}, \mathsf{cq}) = 1$ iff for any honest player i at any rounds r, among any $k = \Omega(\kappa)$ consecutive blocks in the chain $\mathcal{C}_i^r$, the fraction of honest blocks is at least cq. and $r' = r + t$, we have, $\mathsf{len}(\mathcal{C}_i^{r'}) - \mathsf{len}(\mathcal{C}_i^r) \geq t \cdot \mathsf{cg}$.

Definition 10 (Chain quality). *A blockchain protocol $(\Pi^{\mathsf{V}}, \Lambda)$ achieves* chain quality *in Γ_{stat}*-environments *if there exists a chain quality parameter $\mathsf{cq} > 0$ such that*

$$\Pr[\textsc{view} \leftarrow \mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa) : \mathsf{qty}(\textsc{view}, \mathsf{cq}) = 1] \geq 1 - \varepsilon(\kappa),$$

in all Γ_{stat}-admissible environments.

Consistency. For a $\textsc{view} \leftarrow \mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa)$, we define $\mathsf{cns}(\textsc{view}) = 1$ iff for any honest player i at any rounds r and any honest player j at any rounds $r' > r$, after removing the last $k = \Omega(\kappa)$ block of $\mathcal{C}_i^r$, we obtain a prefix of the chain $\mathcal{C}_j^{r'}$.

Definition 11 (Consistency). *A blockchain protocol $(\Pi^{\mathsf{V}}, \Lambda)$ achieves* chain quality *in Γ_{stat}*-environments *if*

$$\Pr[\textsc{view} \leftarrow \mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa) : \mathsf{cns}(\textsc{view}) = 1] \geq 1 - \varepsilon(\kappa),$$

in all Γ_{stat}-admissible environments.

The Nakamoto's protocol achieves chain growth, chain quality, and consistency as follows.

Lemma 11 (Theorem 4.1 (chain growth), Theorem 4.3 (consistency), Theorem 4.2 (chain quality) in [26]). *Consider Nakamoto's protocol in Γ^*_{PSS}-admissible environments. Nakamoto's protocol achieves chain growth with parameter $\mathsf{cg} = (1-\epsilon)\gamma$ where $\epsilon > 0$, $\gamma = \frac{\alpha}{1+\Delta\alpha}$; chain quality with parameter $\mathsf{cq} = 1 - (1+\epsilon)\frac{\beta}{\gamma}$; and consistency.*

B.3 Achieving the security properties from reliable dissemination

We show that if the CoSpaN protocol achieves reliable dissemination in the SNM model, then it achieves the security properties of persistence, liveness, chain growth, chain quality, and consistency. This can be done by demonstrating a reduction from the security of Π_{CSN} in the SNM model to the security of Nakamoto's protocol Π_{Nak} in the PSS model.

Proof of Lemma 2. We will reduce the security of Π_{CSN} in Γ^*_{stat}-admissible environments to the security of Π_{Nak} in Γ^*_{PSS}-admissible environments.

More concretely, consider a tuple $(n, Q, \eta, \rho, \delta, \mathcal{A}, \mathcal{Z})$ that is Γ^*_{stat}-admissible with respect to Π_{CSN}. We will construct an environment $\mathcal{Z}'$ and an adversary $\mathcal{A}'$ such that the tuple $(Q, \rho, \Delta, \mathcal{A}', \mathcal{Z}')$ is Γ^*_{PSS}-admissible with respect to Π_{CSN}. Based on Lemma 1, protocol Π_{Nak} achieves persistence and liveness in Γ^*_{PSS}-admissible environments. Thus, we can prove the security of Π by mapping the execution of protocol Π_{CSN} in $(n, Q, \eta, \rho, \delta, \mathcal{A}, \mathcal{Z})$ to the execution of protocol Π_{Nak} in (Q, ρ, Δ). Indeed, consider the execution $\mathsf{EXEC}^{\Pi_{\mathrm{CSN}}}_{\mathcal{A},\mathcal{Z}}(\kappa)$ and a modified execution in which after some round r, nodes run the protocol Π_{CSN} in the presence of $\mathcal{Z}'$ and $\mathcal{A}'$. We will show that by fixing the randomness, the views in the support of the two executions are *compatible* with high probability.

We say $\textsc{view}_1$ and $\textsc{view}_2$ are *compatible*, denoted by $\textsc{view}_1 \overset{c}{=} \textsc{view}_2$, if at any round r, for any honest node i in $\textsc{view}_1$, there exists an honest node j in $\textsc{view}_2$ such that the local chain of i and j are the same.

We construct an environment $\mathcal{Z}'$ from the environment $\mathcal{Z}$ as follows:

- For each node i that is activated by $\mathcal{Z}$, if $q_i > 0$, the environment $\mathcal{Z}'$ initiates a set $F(i)$ of q_i nodes in which each node has 1 mining power.
- If the environment $\mathcal{Z}$ corrupts a node i, the environment $\mathcal{Z}'$ corrupts all nodes in $F(i)$.
- In each round, the environment $\mathcal{Z}'$ sends the same input that $\mathcal{Z}$ sends to i to all nodes in $F(i)$.
- In each round, nodes in $F(i)$ receive the same results from the random oracle queries as the node i.

We also construct a new adversary $\mathcal{A}'$ from the adversary $\mathcal{A}$ as follows:

- Upon $\mathcal{A}$ sending a message to an honest node i (from a malicious node), the adversary $\mathcal{A}'$ sends the same message to all nodes in $F(i)$.
- Consider a message m that is generated at an honest i. Let $\Delta_{ij}(m)$ be the time it takes for node i to send a message to node j in the point-to-point communication model. Upon a node in $F(i)$ broadcasting the message m, the adversary $\mathcal{A}'$ immediately sends m to all nodes in $F(i)$ and delays for $\min(\Delta, \Delta_{ij}(m))$ rounds before sending the message to all nodes in $F(j)$.

As $\mathcal{A}$ and $\mathcal{Z}$ are probabilistic polynomial-time algorithms, we have that $\mathcal{A}'$ and $\mathcal{Z}'$ are also probabilistic polynomial-time algorithms. Additionally, $\Gamma^*_{\mathrm{stat}}(n, Q, \eta, \rho, \delta) = 1$, so $\Gamma^*_{\mathrm{PSS}}(Q, \rho, \Delta) = 1$, meaning that $(Q, \rho, \Delta, \mathcal{A}', \mathcal{Z}')$ is Γ^*_{PSS}-admissible with respect to Π_{CSN}.

Let $\mathsf{EXEC1}(\lambda)$ and $\mathsf{EXEC2}(\lambda)$ denote the output of the executions $\mathsf{EXEC}_{\Pi_{\mathrm{CSN}},\mathcal{A},\mathcal{Z}}(\kappa)$ and $\mathsf{EXEC}_{\Pi_{\mathrm{Nak}},\mathcal{A}',\mathcal{Z}'}(\kappa)$, respectively, with the randomness fixed to λ. Here, the randomness λ can be partitioned into two parts: λ_1 is used when nodes query the random oracle to generate new blocks, and λ_2 is used when nodes generate the keys of the VRFs.

Let w_i be the latest round in $\mathsf{EXEC1}(\lambda)$ such that the maximum epoch in the view of an honest player is i. Let $\mathsf{EXEC1}_{w_i}(\lambda)$ and $\mathsf{EXEC2}_{w_i}(\lambda)$ be the joint view of all nodes in $\mathsf{EXEC1}(\lambda)$ and $\mathsf{EXEC2}(\lambda)$, respectively, until round w_i.

We consider the case where $\mathsf{rel}(\mathsf{EXEC1}_{w_i}(\lambda), \Delta) = 1$, i.e., any honest node in $\mathsf{EXEC1}_{w_i}(\lambda)$ can disseminate any message to all honest nodes within Δ rounds. In this case, for any honest node i and any message m, the nodes in $A(i)$ (in $\mathsf{EXEC2}_{w_i}(\lambda)$) receive m (from other honest nodes and the adversary) at the same round as the node i (in $\mathsf{EXEC1}_{w_i}(\lambda)$). Thus, at any round r, the local chain of i is the same as the local chain of all nodes in $A(i)$, i.e., $\mathsf{EXEC1}_{w_i}(\lambda) \overset{c}{=} \mathsf{EXEC2}_{w_i}(\lambda)$.

We have,

$$\Pr_{\lambda}[\mathsf{EXEC1}_{w_i}(\lambda) \overset{c}{=} \mathsf{EXEC2}_{w_i}(\lambda)] \geq \Pr_{\lambda}[\mathsf{rel}(\mathsf{EXEC1}_{w_i}(\lambda), \Delta)]. \quad (8)$$

Consider an honest node i at round r, and an honest node j at round $r' \geq r$ in $\mathsf{EXEC1}_{w_i}(\lambda)$. If $\mathsf{EXEC1}_{w_i}(\lambda) \overset{c}{=} \mathsf{EXEC2}_{w_i}(\lambda)$, then there exists an honest node i' at round r and an honest node j' at round r' in $\mathsf{EXEC2}_{w_i}(\lambda)$ such that the local chain of i at round r (respectively, the local chain of j at round r') is the same as the local chain of i' at round r (respectively, the local chain of j' at round r'). Thus, based on Def. 2, if $\mathsf{EXEC1}_{w_i}(\lambda) \overset{c}{=} \mathsf{EXEC2}_{w_i}(\lambda)$, then $\mathsf{per}(\mathsf{EXEC2}_{w_i}(\lambda)) = \mathsf{per}(\mathsf{EXEC1}_{w_i}(\lambda))$. Using similar arguments for liveness and reliable dissemination, we have,

$$\text{If } \mathsf{EXEC1}_{w_i}(\lambda) \overset{c}{=} \mathsf{EXEC2}_{w_i}(\lambda),$$
$$\text{then } \mathsf{sec}(\mathsf{EXEC1}_{w_i}(\lambda)) = \mathsf{sec}(\mathsf{EXEC2}_{w_i}(\lambda)).$$

Hence, we have,

$$\begin{aligned}
&\Pr[\textsc{view} \leftarrow \mathsf{EXEC}^{\Pi_{\text{CSN}}}_{\mathcal{A},\mathcal{Z}}(\kappa) : \mathsf{sec}(\textsc{view}) = 1] \\
=&\Pr_{\lambda}[\mathsf{sec}(\mathsf{EXEC1}(\lambda)) = 1] \\
\geq&\Pr_{\lambda}[\mathsf{sec}(\mathsf{EXEC2}(\lambda)) = 1] \times \Pr_{\lambda}[\mathsf{EXEC1}_{w_i}(\lambda) \overset{c}{=} \mathsf{EXEC2}_{w_i}(\lambda)] \\
\geq&(1 - \varepsilon(\kappa)) \times \Pr_{\lambda}[\mathsf{EXEC1}_{w_i}(\lambda) \overset{c}{=} \mathsf{EXEC2}_{w_i}(\lambda)] \quad \text{(Lemma 1)} \\
\geq&\Pr_{\lambda}[\mathsf{rel}(\mathsf{EXEC1}_{w_i}(\lambda), \Delta)] - \varepsilon(\kappa) \quad \text{(Eq. 8)}.
\end{aligned}$$

□

B.4 Achieving reliable dissemination from the security properties

Given the event that protocol Π_{CSN} achieves the security properties (persistence, liveness, chain growth, chain quality, and consistency) and reliable dissemination up until epoch i, we will prove that protocol Π_{CSN} achieves reliable dissemination up until epoch $i + 1$ with high probability.

Core-periphery graph Given the event that protocol Π_{CSN} achieves the security properties (persistence, liveness, chain growth, chain quality, and consistency) and reliable dissemination up until epoch i, we will prove that protocol Π_{CSN} achieves reliable dissemination up until epoch $i + 1$ with high probability.

Let $\mathsf{G} = (\mathsf{N}, \mathsf{E})$ be the network at the i-th epoch[8], where N is the set of participating nodes and E is the set of connections. The network G can be obtained from the *core-periphery graph* that is constructed using the P2P protocol Π_{P2P}. Let $\bar{\mathsf{G}} = (\bar{\mathsf{N}}, \bar{\mathsf{E}})$ be the core-periphery graph, where $\bar{\mathsf{N}} = \mathsf{N} \cup \bar{\mathsf{C}}$, $\bar{\mathsf{C}}$ is a set of core registrations that are included in the registration period, and $\bar{\mathsf{E}}$ is the set of verifiable random connections that are established in the topology construction of the P2P protocol Π_{P2P}. In the P2P protocol Π_{P2P}, each participating node in N acts as *one periphery node* and (possibly) *multiple core nodes*. For example, if a participating node v has two core registrations included in the registration period, it can be mapped as 3

[8] G is constructed at the end of epoch i and will be used in epoch $i + 1$.

nodes in $\bar{\mathsf{N}}$: one periphery node and two core nodes. Each node in $\bar{\mathsf{G}}$ can be mapped to exactly one node in G. Thus, we can obtain the network G by merging corresponding nodes in the core-periphery graph $\bar{\mathsf{G}}$.

Let $\bar{\mathsf{H}}$ be the set of honest nodes in the core-periphery graph $\bar{\mathsf{G}}$, i.e., the set of core and periphery nodes that are mapped to honest participating nodes in H. Let $\mathsf{G}[\mathsf{H}]$ be the *honest network* with the set of nodes H and the set of edges consisting of all the edges in G that have both endpoints in H. Similarly, let $\bar{\mathsf{G}}[\bar{\mathsf{H}}]$ be the *honest core-periphery graph.* The diameter of the honest network $\mathsf{G}[\mathsf{H}]$ can be upper-bounded by the diameter of the honest core-periphery graph $\bar{\mathsf{G}}[\bar{\mathsf{H}}]$, i.e., $\mathsf{Diam}(\mathsf{G}[\mathsf{H}]) \leq \mathsf{Diam}(\bar{\mathsf{G}}[\bar{\mathsf{H}}])$. Indeed, each node in $\bar{\mathsf{G}}[\bar{\mathsf{H}}]$ can be mapped to exactly one node in $\mathsf{G}[\mathsf{H}]$. Thus, the distance between two honest nodes in $\mathsf{G}[\mathsf{H}]$ is bounded by the distance of the two corresponding nodes in $\bar{\mathsf{G}}[\bar{\mathsf{H}}]$. We show that protocol Π_{CSN} achieves reliable dissemination based on the diameter $\mathsf{Diam}(\bar{\mathsf{G}}[\bar{\mathsf{H}}])$ of the honest core-periphery graph as follows.

Lemma 12. *For any epoch i, we have*

$$\Pr\left[\mathcal{R}_{i+1} \mid \mathcal{S}_i \wedge \mathcal{R}_i\right] \geq \Pr\left[\mathsf{Diam}(\bar{\mathsf{G}}[\bar{\mathsf{H}}]) \leq O(\log n) \mid \mathcal{S}_i \wedge \mathcal{R}_i\right].$$

Proof. We first show that any honest node can deliver any message to all other honest nodes within Δ rounds via the honest network $\mathsf{G}[\mathsf{H}]$. Indeed, a node can send messages to all its neighbors on the honest network $\mathsf{G}[\mathsf{H}]$ within δ rounds. Additionally, the distance between two honest nodes on $\mathsf{G}[\mathsf{H}]$ is at most $\frac{\Delta}{\delta}$. Thus, any message from an honest node can be delivered to all honest nodes within Δ rounds.

Next, we show that nodes can finish establishing connections before epoch $i+1$ starts. Equivalently, we will show that nodes cannot generate more than k blocks in t_{e} rounds.

Let r_1 be the first round in the execution in which the length of the chain of any honest node is at least $i \cdot L - k$, i.e., all honest nodes enter the second part of the grace period of epoch i and start establishing the new connection. Let $r_2 = r_1 + t_{\text{e}}$. From round $r_1 + 1$ to round r_2, nodes make at most $t_{\text{e}} \cdot Q$ queries to the random oracle in which the probability of success in generating a new block of a single query is p. We represent each query by a Bernoulli random variable with an expected value of p. Let n_{b} be the number of blocks that are generated from round $r_1 + 1$ to round r_2. For any $\epsilon > 0$ such that $t_{\text{e}} < \frac{k}{(1+\epsilon)Q \cdot p}$, using the Chernoff bound in Lemma 10 on at most $t_{\text{e}} \cdot Q$ Bernoulli random variables, we have,

$$\Pr[n_{\text{b}} < k] = 1 - e^{-\Omega(\kappa)}.$$

Thus, at round r_2, with a probability of $1 - e^{-\Omega(\kappa)}$, the length of the longest chain is at most $i \cdot L - k + n_{\text{b}} < i \cdot L$, i.e., nodes have not yet entered epoch $i+1$. Hence, the network $\bar{\mathsf{G}}^{(i)}$ is constructed before epoch $i+1$ starts. Additionally, any honest node can deliver any message to all other honest nodes within Δ rounds via the honest network $\mathsf{G}[\mathsf{H}]$. Therefore, protocol Π_{CSN} achieves Δ-reliable dissemination until epoch $i+1$. □

Honest core nodes

Proof of Lemma 3. We first show that with high probability, all honest nodes have the same view of the set of core nodes. Here, we define the set of core nodes $\bar{\mathsf{C}}$ as the union of all the sets of core nodes in the view of honest nodes. For each participating honest node $u \in \mathsf{H}$, let $\bar{\mathsf{C}}^{(u)}$ be the set of core nodes in the view of node u. The set of core nodes is defined as

$$\bar{\mathsf{C}} = \bigcup_{u \in \mathsf{H}} \bar{\mathsf{C}}^{(u)}.$$

In the CoSpaN protocol, nodes start constructing the network after k blocks since the registration period ends. Thus, based on the consistency property, when nodes construct the network, all honest nodes have the same view of the blocks in the registration period. Hence, they can obtain the same set of core nodes $\bar{\mathsf{C}}$. Consequently, we have that all honest nodes have the same view of the set of core nodes $\bar{\mathsf{H}}_{\text{c}}$.

Let r_1 be the first round in the execution in which the length of the chain of any honest node is at least iL, i.e., all honest nodes are at epoch i.

For any $\epsilon_1 > 0$, let $r_2 = r_1 + (1-\epsilon_1)\frac{L_{\mathtt{reg}}-k}{Q\cdot p}$. From round r_1+1 to round r_2, nodes make at most $(1-\epsilon_1)\frac{L_{\mathtt{reg}}-k}{p}$ queries to the random oracle in which the probability of success in generating a new block of a single query is p. We represent each query by a Bernoulli random variable with an expected value of p. Let n_{b} be the number of blocks that are generated from round r_1+1 to round r_2. By using the Chernoff bound in Lemma 10 on at most $(1-\epsilon_1)\frac{L_{\mathtt{reg}}-k}{Q\cdot p}$ Bernoulli random variables, we have,

$$\Pr[n_{\mathrm{b}} < k] \leq 1 - e^{-\Omega(\kappa)}.$$

Using the Chernoff bound in Lemma 10, we have,

$$\Pr[n_{\mathrm{b}} < (1-\epsilon_1)(1+\epsilon_1)(L_{\mathtt{reg}}-k)] 1 - e^{-\Omega(\kappa)}$$
$$\Rightarrow \Pr[n_{\mathrm{b}} < (L_{\mathtt{reg}}-k)] 1 - e^{-\Omega(\kappa)}$$

Let $\bar{h_{\mathrm{c}}}$ be the number of registration blocks generated by honest nodes willing to participate as core nodes from round r_1 to r_2. In each round, honest nodes willing to participate as core nodes make $(1-\rho)\omega$ queries to the random oracle. Thus, from round r_1 to round r_2, honest nodes willing to participate as core nodes make $(1-\epsilon_1)\frac{L_{\mathtt{reg}}-k}{p}(1-\rho)\omega$ queries to the random oracle. Here, the probability of success in generating a new registration block from a single query is $p' = p\frac{s}{L_{\mathtt{reg}}}$. We represent each query by a Bernoulli random variable with an expected value of p'. We choose $\epsilon_2 > 0$ such that $1-\epsilon = (1-\epsilon_1)(1-\epsilon_2)$. Using the Chernoff bound in Lemma 10, we have,

$$\Pr[\bar{h_{\mathrm{c}}} > (1-\epsilon)(1-\rho)(1-\frac{1}{L_{\mathtt{reg}}})\omega \cdot s] > 1 - e^{-\Omega(\kappa)}.$$

At round r_2, with a probability of 1 - $e^{-\Omega(\kappa)}$, we have, 1) the length of the longest chain is at most $iL + L_{\mathtt{reg}} - k$; and 2) the number of registration blocks that are generated by honest nodes, who are willing to participate as core nodes is at least $(1-\epsilon)(1-\rho)(1-\frac{1}{L_{\mathtt{reg}}})\omega$.

From the chain quality property in Lemma 11, with a probability of $1-e^{-\Omega(\kappa)}$, there exists an honest block from the $(i\cdot L + L_{\mathtt{reg}} - \kappa + 1)$-th block to the $(i \cdot L + L_{\mathtt{reg}})$-th block. Therefore, if an honest node registers as a core node (by propagating a registration block) before the $(L_{\mathtt{reg}} - k)$-th block of the registration period is generated, it will be selected as a core node in the next epoch (i.e., the registration block will be included on the blockchain). Thus, we have,

$$\Pr[h_{\mathrm{c}} \geq \bar{h_{\mathrm{c}}}] > 1 - e^{-\Omega(\kappa)}$$
$$\Rightarrow \Pr[h_{\mathrm{c}} \geq \mathsf{q} \cdot s] > 1 - e^{-\Omega(\kappa)}$$

□

Verifiable random connections

Proof of Lemma 4. First, we show that all honest nodes obtain the same randomness rnd_i at epoch i. In the CoSpaN protocol, nodes obtain the randomness rnd_i after k blocks since the registration period ends. Based on the consistency property, at that moment, all honest nodes have the same view of the blocks in the registration period. As the randomness rnd_i is computed based on the last k blocks in the registration period, all honest nodes obtain the same value of the randomness rnd_i.

Let $\bar{X}_{u\to v}$ be the event where a core node u can solve the inequality in Eq.3 for another core or periphery node v. We prove that, for any honest core nodes u, v, we have,

$$\Pr[\bar{X}_{u\to v}] \geq \bar{\mu} = \frac{d}{2s} - \varepsilon(\kappa).$$

Assume towards contradiction that there exists a non-negligible number p such that

$$\Pr[\bar{X}_{u\to v}] < \frac{d}{2s} - p.$$

Let sk' be the private key for VRF of the node u. Since the public-key encryption and the signature schemes are secure, we have,

$$\Pr\left[F_{\mathsf{sk}'}(\mathsf{rnd}_i \| v) < \frac{d}{2s}2^{\kappa}\right] < \frac{d}{2s} - p$$

We will construct an adversary $\mathcal{A}$ that is given a query to the oracle $\mathsf{Prove}_{\mathsf{sk}_v}$ as follows.

- Compute $x = \mathsf{rnd}_i \| v$ and send x to the prover.
- Upon receiving y_b from the prover.
- If $y_b < \frac{d}{2s}2^{\kappa}$, return $b' = 1$. Otherwise, return $b' = 0$.

We have

$$\begin{aligned}\Pr[b = b'] &= \frac{1}{2}\left(\Pr\left[y_0 \geq \frac{d}{2s}2^{\kappa}\right] + \Pr\left[y_1 < \frac{d}{2s}2^{\kappa}\right]\right)\\ &= \frac{1}{2}\left(\Pr\left[F_{\mathsf{sk}'}(x) \geq \frac{d}{2s}2^{\kappa}\right] + \frac{d}{2s}\right)\\ &> \frac{1}{2}\left(1 - \frac{d}{2s} + p + \frac{d}{2s}\right)\\ &= \frac{1}{2}(1+p)\end{aligned}$$

This contradicts the pseudo-randomness property of VRF.

Similarly, for any honest core node u and a periphery node v, we have,

$$\Pr[\bar{X}_{u\to v}] \geq \bar{\mu}.$$

The node v accepts the connection for a core node u if node u can provide an output of the VRF that satisfies the hash inequality in Eq. 3. If the node u is honest, it always requests to establish a connection to a node v if it can find the suitable VRF output. Hence, we have,

$$\Pr[X_{u\to v}] = \Pr[\bar{X}_{u\to v}] \geq \bar{\mu}.$$

□

Achieving reliable dissemination We now prove that the protocol Π_{CSN} can achieve reliable dissemination by showing that the diameter of the honest core-periphery graph $\bar{\mathsf{G}}[\bar{\mathsf{H}}]$ is bounded. We bound the diameter $\mathsf{Diam}(\bar{\mathsf{G}}[\bar{\mathsf{H}}])$ in two steps as follows. First, we bound the diameter of the network of honest core nodes $\bar{\mathsf{G}}[\bar{\mathsf{H}}_{\mathrm{c}}]$ (Lemma 13). Secondly, we show that all periphery nodes are connected with at least one honest core node (Lemma 14).

Lemma 13. *For $d \geq 4\log s$, we have,*

$$\Pr[\mathsf{Diam}(\bar{\mathsf{G}}[\bar{\mathsf{H}}_{\mathrm{c}}]) \leq O(\log n)] \geq 1 - e^{-\Omega(\kappa)}.$$

Proof. For any honest core nodes $u, v \in \bar{\mathsf{H}}_{\mathrm{c}}$, we have,

$$A_{uv} \geq \mu,$$

where $\mu = 1 - (1 - \bar{\mu})^2$, $\bar{\mu} = \frac{d}{2s} - \varepsilon(\kappa)$.

Thus, there exists a constant $c_1 > 2$ such that

$$\mu = \frac{c_1 \cdot \log h_{\mathrm{c}}}{h_{\mathrm{c}}}.$$

For any node $v \in \bar{\mathsf{H}}_{\mathrm{c}}$, let

$$N_k(v) = \{w \in \bar{\mathsf{H}}_{\mathrm{c}} : \mathit{dist}(v, w) = k\},$$

where $\mathit{dist}(v, w)$ is the distance between v and w. We define the event

$$F_k = |N_k(v)| \in I_k = \left[\left(\frac{h_{\mathrm{c}} \cdot \mu}{2} \right)^k, (2h_{\mathrm{c}} \cdot \mu)^k \right].$$

For any constant $c' > 0$, let $q = \log(h_{\mathrm{c}}) * \log(h_{\mathrm{c}}^2/c') / \log(h_{\mathrm{c}} \cdot \mu)$. Let $\mathsf{bin}(x, y)$ be the binomial distribution with parameters x and y, i.e., the discrete probability distribution of the number of successes in a sequence of x independent experiments in which the probability of success in each experiment is y. For $k \leq \lceil q/2 \rceil$, we have

$$\begin{aligned}
&\Pr[\bar{F_k} \mid \ F_1, \cdots, F_{k-1}] \\
=&\Pr\left[\mathsf{bin}\left(h_{\mathrm{c}} - \sum_{i=1}^{k-1} |N_i(v), 1 - (1-p)^{|N_{k-1}(v)|}\right) \notin I_k\right] \\
\leq&\Pr\left[\mathsf{bin}\left(h_{\mathrm{c}} - o(h_{\mathrm{c}}), \frac{3}{4}\left(\frac{h_{\mathrm{c}} \cdot \mu}{2}\right)^{k-1} \mu\right) \leq \left(\frac{h_{\mathrm{c}} \cdot \mu}{2}\right)^k\right] \\
&\quad + \Pr\left[\mathsf{bin}\left(h_{\mathrm{c}} - o(h_{\mathrm{c}}), \frac{5}{4}\left(2h_{\mathrm{c}} \cdot \mu\right)^{k-1} \mu\right) \geq \left(2h_{\mathrm{c}} \cdot \mu\right)^k\right] \\
=& \, e^{-\Omega((h_{\mathrm{c}} \cdot \mu)^k)} = e^{-\Omega(d^k)}.
\end{aligned}$$

Using union bound, we have

$$\begin{aligned}
\Pr[\bar{F_k}] &\leq \sum_{i=1}^{k} \Pr[\bar{F_i} \mid \ F_1, \cdots, F_{i-1}] \\
&= \sum_{i=1}^{k} e^{-\Omega(d^k)} = e^{-\Omega(d)}.
\end{aligned}$$

Thus, with probability $1 - e^{-\Omega(d)}$, for any node u, v, we have

$$\begin{aligned}
|N_{\lceil \log(h_c)/2 \rceil}(v)| &\geq \left(\frac{h_{\mathrm{c}} \cdot \mu}{2}\right)^{\lceil \log(h_c)/2 \rceil}, \\
\text{and } |N_{\lfloor \log(h_c)/2 \rfloor}(u)| &\geq \left(\frac{h_{\mathrm{c}} \cdot \mu}{2}\right)^{\lfloor \log(h_c)/2 \rfloor}.
\end{aligned}$$

Let $X = N_{\lceil \log(h_c)/2 \rceil}(v)$ and $Y = N_{\lfloor \log(h_c)/2 \rfloor}(u) \neq \emptyset$ We have, either $X \cap Y \neq \emptyset$ or

$$\Pr[\nexists \text{ an edge between } X, Y] = (1 - \mu)^{\left(\frac{h_{\mathrm{c}} \cdot \mu}{2}\right)^{\log(h_c)}}$$

$$\leq e^{-\Omega(d)}.$$

In other word, for any node u, v, we have

$$\Pr[dist(u,v) > \log(h_c) + 1] \leq e^{-\Omega(d)}.$$

Using union bound on all h_c^2 pairs of nodes, we have,

$$\begin{aligned}\Pr[\mathsf{Diam}(\bar{\mathsf{G}}[\bar{\mathsf{H}}_c]) > \log(h_c) + 1] &\leq h_c^2 e^{-\Omega(d)} = e^{-\Omega(d) - \log h_c} \\ &= e^{-\Omega(d)}.\end{aligned}$$

□

Lemma 14. *Let* RB *be the event where all honest periphery nodes have at least one connection to a core node. For* $d > \frac{k}{-\log \mathsf{q}}$*. We have,*

$$\Pr[\mathsf{RB} \mid \mathcal{S}_i \wedge \mathcal{R}_i] \geq 1 - e^{-\Omega(\kappa)}.$$

Proof. Recall that, the probability that there exists a connection between an honest core and an honest periphery node is $\frac{d}{2s}$. Thus, the probability that an honest periphery node has no connection to any honest core nodes is

$$\begin{aligned}\left(1 - \frac{d}{2s}\right)^{h_c} &= \left(\left(1 - \frac{d}{2s}\right)^{\frac{2s}{d}}\right)^{\frac{h_c \cdot d}{2s}} \\ &\leq e^{-\frac{h_c \cdot d}{2s}}\end{aligned}$$

Since the number of periphery honest node is smaller than h, using union bound, we have, the probability that there exists an honest periphery node has no connection to any honest core nodes is

$$\begin{aligned}n \cdot e^{-\frac{h_c d}{2s}} &= e^{-\left(\frac{h_c d}{2s} - \log h\right)} \\ &= e^{-\Omega(\kappa)}.\end{aligned}$$

□

We are now ready to prove Lemma 5.

Proof of reliable dissemination (Lemma 5). Combining Lemma 13 and Lemma 14, with probability $1 - e^{-\Omega(\kappa)}$, we have that the maximum distance between two honest core nodes is at most $\log h_c + 1$, and all honest periphery nodes connect to at least one honest core node. Thus, the maximum distance between an honest periphery node and an honest core node is at most $\log h_c + 2$. Therefore, the maximum distance between two honest periphery nodes is at most $\log h_c + 3$.

We have that

$$\Pr[\mathsf{Diam}(\bar{\mathsf{G}}^{(i)}) \leq \log h_c + 3] \geq 1 - e^{-\Omega(d)}.$$

Note that an honest node can send any messages to all of its neighbors within a network round of δ_l time units. Thus, with a probability of $1 - e^{-\Omega(d)}$, an honest node can send any messages to all honest nodes within

$$\Delta = O(\log h_c) \times \delta_l.$$

□

B.5 Network sparsity

Proof of Lemma 6. We first analyze the sparsity of the core-periphery graph $\bar{\mathsf{G}}$ and the network G. To analyze the sparsity of the core-periphery graph, we analyze the maximum number of core nodes and then bound the number of connections from those core nodes.

We bound the number of connection in the core-periphery graph when the randomness rnd_i is fixed. Let r_1 be the first round in the execution in which the length of the chain of any honest node is at least iL, i.e., all honest nodes are at epoch i. For any $\epsilon_1 > 0$, let $r_2 = r_1 + (1+\epsilon_1)\frac{L_{\mathtt{reg}}}{Qp}]$. Let n_{b}, n_{c} be the number of blocks, registration blocks that are generated from round $r_1 + 1$ to round r_2.

From round $r_1 + 1$ to round r_2, nodes make at most $(1+\epsilon_1)\frac{L_{\mathtt{reg}}-k}{p}$ queries to random oracle in which the probability of success in generating a new block, registration block of a single query are p, p' respectively.

We choose ϵ_2 such that $(1-\epsilon_2)(1+\epsilon_1) = 1$, using the Chernoff bound in Lemma 10 on at most $(1+\epsilon_1)\frac{L_{\mathtt{reg}}-k}{p}$ Bernoulli random variables with the expected value of p, we have,

$$\Pr[n_{\mathrm{b}} \geq L_{\mathtt{reg}} - k] \geq 1 - e^{-\Omega(\kappa)}.$$

We choose $\epsilon_3, \epsilon_4 > 0$ such that $1+\epsilon_4 = (1+\epsilon_1)(1+\epsilon_3)$. Using the Chernoff bound in Lemma 10 on at most $(1+\epsilon_1)\frac{L_{\mathtt{reg}}-k}{p}$ Bernoulli random variables with the expected value of $p' = p\frac{s}{L_{\mathtt{reg}}}$, we have,

$$\Pr[n_{\mathrm{b}} \geq (1+\epsilon_4)s] \geq 1 - e^{-\Omega(\kappa)}.$$

At round r_2, with a probability of 1 - $e^{-\Omega(\kappa)}$, we have, 1) the length of the longest chain is at least $iL + L_{\mathtt{reg}}$; and 2) the number of registration blocks is at most $(1+\epsilon_4)s$. Recall that, in the CoSpaN protocol, nodes can only register as core nodes in the registration period. Thus, with a probability of 1 - $e^{-\Omega(\kappa)}$, the number of core nodes at epoch i is at most $\hat{s} = (1+\epsilon_4)s$.

For a core node v and a (core or periphery) node u, let $X_{v\to u}$ be the Bernoulli random variable that represent whether or not there exists a connection from v to u. We have, $\Pr[X_{v\to u} = 1] = \bar{\mu} = \frac{d}{2s}$.

In the the core-periphery graph $\bar{\mathsf{G}}$, with a probability of 1 - $e^{-\Omega(\kappa)}$, there are at most n periphery nodes and $\hat{s}$ core nodes. Thus, the number of connections in G_+ is at most

$$\sum_{v=1}^{\hat{s}}\sum_{u=1}^{n+\hat{s}} X_{v\to u}.$$

Using the Chernoff bound in Lemma 10, for $\epsilon_5 > 0$, we have,

$$\Pr[\sum_{v=1}^{\hat{s}}\sum_{u=1}^{n} X_{v\to u} \leq (1+\epsilon_5)\hat{s}(n+\hat{s})\frac{d}{2s}] \geq 1 - e^{-\Omega((n+s)\frac{d}{2})}.$$

Hence, there exists a constant $\epsilon > 0$ such that, with a probability of 1 - $e^{-\Omega(\kappa)}$, the number of connections in G_+ is at most $(1+\epsilon)(n+s)\frac{d}{2}$.

Now, we analyze degree of a node when the randomness rnd is computed based on the hash value of the blocks in the previous epoch. Since the miner does not know the hash value until the new block is created, we consider the hash value of *any block* as a perfect randomness. In other words, the adversary cannot predict the hash value of future blocks (even when that block is create by a malicious miner). Thus, the adversary can only manipulate by changing the hash value of the last block in the registration period, i.e., if an malicious miner create the last block in the registration period, the adversary can choose either to publish the that block or not. Thus, the adversary can try at most $O(\kappa)$ different values of the randomness rnd. Using union bound, we have with probability at least $1 - \kappa e^{\Omega((n+s)\frac{d}{2} - \log k)} = 1 - e^{\Omega(\kappa)}$, for any constant $\epsilon > 0$, the number of connections in G_+ is at most $(1+\epsilon)(n+s)\frac{d}{2}$.

We now bound the sparsity of the network G based on the core-periphery graph $\bar{\mathsf{G}}$. Since the sum of degree of nodes in network G does not exceed that in the the core-periphery graph $\bar{\mathsf{G}}$, with a probability of

1 - $e^{-\Omega(\kappa)}$, the number of connections in G is at most $(1+\epsilon)(n+s)\frac{d}{2}$. Thus, the sum of degree of nodes in G is

$$(1+\epsilon)(n+s)d.$$

As the number of honest nodes is at least $n(1-\eta)$, with a probability of 1 - $e^{-\Omega(\kappa)}$the average degree of nodes in G is

$$(1+\epsilon)\frac{(n+s)d}{n(1-\eta)}(1+\epsilon) = (1+\frac{s}{n})/(1-\eta)d.$$

□

B.6 Proof of the main theorem

We are now ready to prove that protocol Π_{CSN} achieves the security properties (Theorem 1).

Proof of Theorem 1. We prove the security of protocol Π_{CSN}, i.e.,

$$\Pr[\text{VIEW} \leftarrow \mathsf{EXEC}^{\Pi_{\text{CSN}}}_{\mathcal{A},\mathcal{Z}}(\kappa) : \mathsf{sec}(\text{VIEW}) = 1] \geq 1 - \varepsilon(\kappa),$$

by induction. For each epoch $j \in [1..e_{\max}]$, we prove the two following hypothesis

$$\Pr[\mathcal{S}_j \wedge \mathcal{R}_{j+1}] > 1 - (2j+1)\varepsilon(\kappa).$$

As the number of epoch in the protocol execution is at most $e_{\max}$, we can conclude that protocol Π_{CSN} is secure if the hypothesis is true for $j = 2e_{\max}$, i.e.,

$$\Pr[\mathcal{S}_{e_{\max}} \wedge \mathcal{R}_{e_{\max}+1}] > 1 - (2e_{\max}+1)\varepsilon(\kappa).$$

Basis step. From the bootstrapping condition, protocol Π_{CSN} achieve reliable dissemination at epoch 1, i.e.,

$$\Pr[\mathcal{R}_1] \geq 1 - \varepsilon(\kappa).$$

Induction step. Given that the hypothesis is true for $j = i-1$, (where $i \in [1..e_{\max}]$), we will prove the hypothesis is true of $j = i$. The induction step consists of two parts as follows.

– *Step i-A:* Given from step $(i-1)$-B that

$$\Pr[\mathcal{S}_{i-1} \wedge \mathcal{R}_i] > 1 - (2i-1)\varepsilon(\kappa).$$

From Lemma 2, we have,

$$\begin{aligned}\Pr[\mathcal{S}_i \wedge \mathcal{R}_i] &\geq \Pr[\mathcal{S}_i \wedge \mathcal{R}_i \mid \mathcal{S}_{i-1} \wedge \mathcal{R}_i] \times \Pr[\mathcal{S}_{i-1} \wedge \mathcal{R}_i] \\ &= \Pr[\mathcal{S}_i \mid \mathcal{S}_{i-1} \wedge \mathcal{R}_i] \times \Pr[\mathcal{S}_{i-1} \wedge \mathcal{R}_i] \\ &> (1-(2i-1)\varepsilon(\kappa))(1-\varepsilon(\kappa)) \\ &> 1 - 2i\varepsilon(\kappa).\end{aligned}$$

– *Step i-B:* Given from step i-A that

$$\Pr[\mathcal{S}_i \wedge \mathcal{R}_i] > 1 - 2i\varepsilon(\kappa).$$

From Lemma 5, we have,

$$\begin{aligned}\Pr[\mathcal{S}_i \wedge \mathcal{R}_{i+1}] &\geq \Pr[\mathcal{S}_i \wedge \mathcal{R}_{i+1} \mid \mathcal{S}_i \wedge \mathcal{R}_i] \times \Pr[\mathcal{S}_i \wedge \mathcal{R}_i] \\ &= \Pr[\mathcal{R}_{i+1} \mid \mathcal{S}_i \wedge \mathcal{R}_i] \times \Pr[\mathcal{S}_1 \wedge \mathcal{R}_i] \\ &> (1-2i\varepsilon(\kappa))(1-\varepsilon(\kappa)) \\ &> 1 - (2i+1)\varepsilon(\kappa).\end{aligned}$$

Thus, we have,

$$\begin{aligned}&\Pr[\text{VIEW} \leftarrow \mathsf{EXEC}^{\Pi_{\text{CSN}}}_{\mathcal{A},\mathcal{Z}}(\kappa) : \mathsf{sec}(\text{VIEW}_i) = 1]\\ \geq&1 - 2i\varepsilon(\kappa).\end{aligned}$$

Since the number of rounds in $\mathsf{EXEC}^{\Pi_{\text{CSN}}}_{\mathcal{A},\mathcal{Z}}(\kappa)$ is polynomial in κ, the number of epochs in $\mathsf{EXEC}^{\Pi_{\text{CSN}}}_{\mathcal{A},\mathcal{Z}}(\kappa)$ is also polynomial in κ. In other words, there exists a polynomial $\mathsf{poly}()$ such that $e_{\max} = \mathsf{poly}(\kappa)$. We have

$$\begin{aligned}&\Pr[\text{VIEW} \leftarrow \mathsf{EXEC}^{\Pi_{\text{CSN}}}_{\mathcal{A},\mathcal{Z}}(\kappa) : \mathsf{sec}(\text{VIEW}) = 1]\\ =&\Pr[\text{VIEW} \leftarrow \mathsf{EXEC}^{\Pi_{\text{CSN}}}_{\mathcal{A},\mathcal{Z}}(\kappa) : \mathsf{sec}(\text{VIEW}_{e_{\max}}) = 1]\\ \geq&1 - 2e_{\max}\varepsilon(\kappa) = 1 - 2\mathsf{poly}(\kappa)\varepsilon(\kappa) = 1 - \varepsilon'(\kappa),\end{aligned}$$

where $\varepsilon'(\cdot)$ is a negligible function.

Further, from Lemma 6, the average degree of a node in each epoch is at most D. Recall that, each epoch consists of L blocks. Thus, for any $\epsilon > 0$, with probability $1 - e^{\Omega(\kappa)}$, each epoch lasts for at least $t_s = \frac{L}{(1+\epsilon)(\alpha+\beta)}$ rounds. In t_s rounds, the length of the chain of honest players increase by at most L. Thus, for any t_s consecutive rounds, there is no overlap between the connections in $\mathsf{G}^{(j)}$ and $\mathsf{G}^{(j+2)}$ ($j \in \mathbb{N}$). Hence, protocol Π_{CSN} achieves $2D$-sparsity. □

C Supplemental for Section 5

C.1 M-adaptive security

We show that the CoSpaN protocol achieves reliable dissemination in the presence of an M-adaptive adversary.

Admissible environments

Definition 12 ($\Gamma_{\text{M-adap}}$-admissible environments). *We say a tuple $(n, Q, \eta, \rho, \delta, \tau_{\text{corr}}, \mathcal{A}, \mathcal{Z})$ with $\Gamma_{\text{M-adap}}(n, Q, \eta, \rho, \delta, \tau_{\text{corr}}$ 1 is $\Gamma_{\text{M-adap}}$-admissible w.r.t $(\Pi^{\mathsf{V}}, \Lambda)$ if $\mathcal{A}$, $\mathcal{Z}$ are probabilistic polynomial-time algorithms, and for every* VIEW *in the support of $\mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa)$, the following holds:*

1. *In each round, the environment $\mathcal{Z}$ can adaptively corrupt additional nodes as long as the number and the hash power of malicious nodes do not exceed ηn and ρQ, respectively. The corruption of an honest node will take τ_{corr} rounds to complete.*
2. *Conditions (2), (3), (4) in Def. 1.*

Proof of Theorem 2

Proof. From the chain growth property, in τ_{corr} rounds, the length of the chain of honest players increases by at least $2L$ blocks. In other words, it takes at least 2 epochs for the adversary to corrupt honest nodes. Thus, to corrupt some honest nodes at epoch i, the adversary must start the corruption before epoch $i-2$. In the execution of protocol Π_{CSN}, the network topology is unpredictable until the core nodes are selected. As the core nodes of epoch i are selected at epoch $i-1$, the adversary cannot predict the network topology of epoch i at epoch $i-2$. Hence, we can achieve the same security as in the static case in Theorem 1. □

C.2 S-adaptive security

We show that the CoSpaN protocol achieves reliable dissemination in the presence of an S-adaptive adversary.

Restricted point-to-point communication model. To allow a slow observation of connections, we introduce a restricted point-to-point communication model in which the environment is responsible for delivering all messages. The deliver delay is determined by an algorithm $\mathcal{D}$ that is proposed by the adversary $\mathcal{A}$ before the execution starts.

Delivering message. Consider at round r, an honest node u send a message to a neighbor v. At a round $r' \in [r+1..r+\delta]$, the environment $\mathcal{Z}$ takes the view of all nodes at the current round and uses algorithm $\mathcal{D}$ to determine whether or not to send the message to node v at the current round. At round $r+\delta$, if the node v have not received the message yet, the environment $\mathcal{Z}$ will send the message to node v.

Observing connection. The adversary can observe a connection after τ_{obs} rounds. Specifically, consider a connection between two nodes u and v that is established at round r. At round $r + \tau_{\text{obs}}$, if u and v are still connected, the adversary knows about the existence of the connection between u and v.

Admissible environments

Definition 13 ($\Gamma_{\text{S-adap}}$-admissible environments). *We say a tuple $(n, Q, \eta, \rho, \delta, \tau_{\text{obs}}, \mathcal{A}, \mathcal{Z})$ with $\Gamma_{\text{S-adap}}(n, Q, \eta, \rho, \delta, \tau_{\text{obs}})$ = 1 is $\Gamma_{\text{S-adap}}$-admissible w.r.t $(\Pi^{\mathsf{V}}, \Lambda)$ if $\mathcal{A}$, $\mathcal{Z}$ are probabilistic polynomial-time algorithms, and for every* VIEW *in the support of $\mathsf{EXEC}_{\Pi^{\mathsf{V}},\mathcal{A},\mathcal{Z}}(\kappa)$, the following holds:*

1. *In each round, the environment $\mathcal{Z}$ can adaptively instantly corrupt additional nodes as long as the number and the hash power of malicious nodes do not exceed ηn and ρQ, respectively.*
2. *Conditions (2), (3) in Def. 1.*
3. *The adversary can observe new connections among nodes after τ_{obs} rounds.*

Honest core nodes

Proof of Lemma 7. Let H be the set of honest nodes. Consider a set $W \subseteq \mathsf{H}$ in which the total mining power of all nodes in W is at least $1 - 2\rho$.

Let r_1 be the first round in the execution in which the length of the chain of any honest node is at least iL, i.e., all honest nodes are at epoch i.

For any $\epsilon_1 > 0$, let $r_2 = r_1 + (1-\epsilon_1)\frac{L_{\texttt{reg}}-k}{Q\cdot p}$. From round r_1+1 to round r_2, nodes make at most $(1-\epsilon_1)\frac{L_{\texttt{reg}}-k}{p}$ queries to random oracle in which the probability of success in generating a new block of a single query is p. We represent each query by a Bernoulli random variable with an expected value of p. Let n_{b} be the number of blocks that are generated from round r_1+1 to round r_2. By using the Chernoff bound in Lemma 10 on at most $(1-\epsilon_1)\frac{L_{\texttt{reg}}-k}{Q\cdot p}$ Bernoulli random variables, we have,

$$\Pr[n_{\text{b}} < k] 1 - e^{-\Omega(\kappa)}.$$

Using the Chernoff bound in Lemma 10, we have,

$$\Pr[n_{\text{b}} < (1-\epsilon_1)(1+\epsilon_1)(L_{\texttt{reg}} - k)] 1 - e^{-\Omega(\kappa)},$$
$$\Rightarrow \Pr[n_{\text{b}} < (L_{\texttt{reg}} - k)] 1 - e^{-\Omega(\kappa)}.$$

Let $\bar{h}_{\text{c}}$ be the number of registration blocks that are generated by the honest nodes that are willing to participate as core nodes from round r_1 to r_2. In each round, the honest nodes in W make at least $(1-2\rho)\omega$ queries to the random oracle. Thus, from round r_1 to round r_2, the honest nodes in W make at least $(1-\epsilon_1)\frac{L_{\texttt{reg}}-k}{p}(1-2\rho)\omega$ queries to the random oracle. Here, the probability of success in generating a new registration block of a single query is $p' = p\frac{s}{L_{\texttt{reg}}}$. We represent each query by a Bernoulli random variable with an expected value of p'. We choose $\epsilon_2 > 0$ such that $1 - \epsilon = (1-\epsilon_1)(1-\epsilon_2)$. Using the Chernoff bound in Lemma 10, we have,

$$\Pr[\bar{h}_{\text{c}} > (1-\epsilon)(1-2\rho)(1-\frac{1}{L_{\texttt{reg}}})\omega s] > 1 - e^{-\Omega(\kappa)}.$$

At round r_2, with a probability of 1 - $e^{-\Omega(\kappa)}$, we have, 1) the length of the longest chain is at most $iL + L_{\texttt{reg}} - k$; and 2) the number of registration blocks that are generated by honest nodes in W who are willing to participate as core nodes is at least $(1-\epsilon)(1-2\rho)(1-\frac{1}{L_{\texttt{reg}}})\omega$.

From chain quality property in Lemma 11, with probability $1 - e^{-\Omega(\kappa)}$, there exist an honest block from the $(iL + L_{\texttt{reg}} - \kappa + 1)$-th blocks to the $(iL + L_{\texttt{reg}})$-th blocks. Thus, if an honest node register as a core node (by propagating a registration block) before the $(L_{\texttt{reg}} - k)$-th block of the registration period is generated, it will be selected as a core node in the next epoch (i.e., the registration block is included on the blockchain). Thus, we have,

$$\Pr[h_c \geq \bar{h_c}] > 1 - e^{-\Omega(\kappa)},$$
$$\Rightarrow \Pr[h_c \geq \mathsf{q}_{\text{M-adap}} \cdot s] > 1 - e^{-\Omega(\kappa)}.$$

□

Verifiable random connection We first prove that the connection among honest nodes are private.

Proof of Lemma 8. We consider two cases of node u as follows.

The node u is a core node (core-core connection). Assume toward contradiction that there exists an adversary $\mathcal{A}$ that can break the security definition in Eq. 5. Let $\mathsf{sk}', \mathsf{pk}'$ be the key pair that the node v holds. Recall that, the node v broadcast an encryption using the public key of the node u. Hence, the adversary cannot learn any information, including the VRF output, from the encryption.

We construct the adversary $\mathcal{A}'$ that is given query to the oracle $\mathsf{Prove}_{\mathsf{sk}'}$ as follows.

– Run $\mathcal{A}$ and obtain u and v from $\mathcal{A}$.
– Compute $x = \mathsf{rnd}_i \| u$ and send x to the prover.
– Upon receiving y_b from the prover.
– If $y_b < 2^\kappa \frac{d}{2s}$, return $b = b'$. Otherwise, return $b = 1 - b'$.

We have,

$$\begin{aligned}
&\Pr[b \neq X_{v \to u}] \\
&= \frac{d}{2s} \Pr[b = 0 | X_{v \to u} = 0] + (1 - \frac{d}{2s}) \Pr[b = 1 | X_{v \to u} = 1] \\
&= \frac{d}{2s} \Pr[b = 0 | F_{\mathsf{sk}_v}(x) < \frac{d}{2s} 2^\kappa] \\
&\quad + (1 - \frac{d}{2s}) \Pr[b = 1 | F_{\mathsf{sk}_v}(x) \geq \frac{d}{2s} 2^\kappa].
\end{aligned}$$

Recall that, as $\mathcal{A}$ that can break the security definition in Eq. 5, there exists a non-negligible number p such that

$$\Pr[X_{v \to u} = 1] > \frac{d}{2s} + p,$$
$$\Rightarrow \Pr[F_{\mathsf{sk}_v}(\mathsf{rnd}_i \| v) < 2^\kappa \frac{d}{2s}] > \frac{d}{2s} + p.$$

Thus, we have

$$\begin{aligned}
\Pr[b = b'] &= \frac{1}{2} \Pr[F_{\mathsf{sk}_v}(\mathsf{rnd}_i \| v) < 2^\kappa \frac{d}{2s} | b = 0] + \\
&\quad \frac{1}{2} \Pr[F_{\mathsf{sk}_v}(\mathsf{rnd}_i \| v) \geq 2^\kappa \frac{d}{2s} | b = 1] \\
&\geq \frac{1}{2} \left(\frac{d}{2s} + p \right) + \frac{1}{2} \left(1 - \frac{d}{2s} \right)
\end{aligned}$$

$$= \frac{1}{2} + \frac{1}{2}p.$$

This contradicts the pseudorandomness property of VRF.

The node u is a periphery node (core-periphery connection). Here, the v directly send a request to connect to node u. Thus, the adversary cannot learn the VRF output that node v sends to node u. Similar to the core-core connection, can prove the core-periphery connection between two honest nodes is private.

□

We are now ready to prove the verifiable random connection.

Proof. We assume toward contradiction that there exists an honest node $u \in \bar{\mathsf{H}} \setminus \bar{\mathsf{W}}$ and an honest core node $v \in \bar{\mathsf{H}}_{\mathrm{c}} \setminus \bar{\mathsf{W}}_{\mathrm{c}}$ and a non-negligible number p such that

$$\Pr_{v \leftarrow \bar{\mathsf{H}}_{\mathrm{c}} \setminus \bar{\mathsf{W}}_{\mathrm{c}}, u \leftarrow \bar{\mathsf{H}} \setminus \bar{\mathsf{W}}}[X_{v \to u} = 1 \mid \mathcal{S}_i \wedge \mathcal{R}_i] < \frac{d}{2s} - p.$$

Recall from Lemma 4 that

$$\Pr_{v \leftarrow \bar{\mathsf{H}}_{\mathrm{c}}, u \leftarrow \bar{\mathsf{H}}}[X_{v \to u} = 1 \mid \mathcal{S}_i \wedge \mathcal{R}_i] \geq \bar{\mu}.$$

Thus, there exists a pair of nodes $v' \in \bar{\mathsf{H}}_{\mathrm{c}}, u' \in \bar{\mathsf{H}}$ in which $v' \in \bar{\mathsf{W}}_{\mathrm{c}}$ and $u' \in \bar{\mathsf{W}}$ such that

$$\Pr[X_{v' \to u'} = 1 \mid \mathcal{S}_i \wedge \mathcal{R}_i] > \frac{d}{2s} + p',$$

where p' is a non-negligible number. This contradicts the private connection property in Lemma 8.

□

C.3 S-adaptive security with mining power concentration

Definition 14 ($\mathit{\Gamma}_{\mathrm{conc}}$-admissible environments). *We say a tuple $(n, Q, \eta, \rho, \delta, \tau_{\mathrm{obs}}, \phi, \gamma, \mathcal{A}, \mathcal{Z})$ with $\mathit{\Gamma}_{\mathrm{conc}}(n, Q, \eta, \rho, \delta, \tau_{\mathrm{obs}},$ 1 is $\mathit{\Gamma}_{\mathrm{conc}}$-admissible w.r.t $(\mathit{\Pi}^{\mathsf{V}}, \mathit{\Lambda})$ if $\mathcal{A}$, $\mathcal{Z}$ are probabilistic polynomial-time algorithms, and for every* VIEW *in the support of $\mathsf{EXEC}_{\mathit{\Pi}^{\mathsf{V}}, \mathcal{A}, \mathcal{Z}}(\kappa)$, the following holds:*

1. *Conditions (1), (2), (3), (4) in Def. 13.*
2. *The top ϕ fraction honest nodes control at least a fraction γ of mining power*

Consider admissible environments with a predicate $\mathit{\Gamma}^*_{\mathrm{conc}}$ in which the honest majority assumption holds. Now we say it holds that $\mathit{\Gamma}^*_{\mathrm{conc}}(n, Q, \eta, \rho, \delta, \tau_{\mathrm{obs}}, \phi, \gamma) = 1$ if for all $n, Q, \delta, \tau_{\mathrm{obs}} \in \mathbb{N}$, $\eta, \rho, \phi, \gamma \in (0, 1)$, there exists $k = \mathit{\Omega}(\kappa)$ such that

$$\mathit{\Gamma}^*_{\text{S-adap}}(n, Q, \eta, \rho, \delta, \tau_{\mathrm{obs}}) = 1 \quad \text{and} \quad \gamma > \rho.$$

As the mining power is concentrated in a small fraction of nodes, the nodes that are selected as core control a large fraction of mining power. We set the core width such that top ϕ fraction of nodes are selected as core nodes. In other words, the set of core nodes control at least a fraction γ of mining power. Since the adversary can not control more than a fraction ρ of mining powers, it cannot corrupt all core nodes. Thus, we can show that the number of honest core nodes is still big enough.

Lemma 15 (Honest core nodes in S-adaptive setting with mining power concentration). *For any $\epsilon > 0$, let $\mathsf{q}_{\mathrm{conc}} = (\gamma - \rho)\omega - \frac{k}{L_{\mathrm{reg}}} - \epsilon$. Consider protocol $\mathit{\Pi}$ with parameter $s > k + \frac{2(1+\epsilon)}{\mathsf{q}_{\mathrm{conc}} \cdot \epsilon^2 \log_2 e} h\phi$, we have,*

$$\Pr[h_{\mathrm{c}} \geq s \cdot \mathsf{q}_{\mathrm{conc}} \mid \mathcal{S}_i \wedge \mathcal{R}_i] > 1 - e^{-\mathit{\Omega}(\kappa)}.$$

Proof. Let K be the set of the top honest nodes that control at least a fraction γ of honest mining power. Consider a set $W \subseteq K$ in which the total mining power of all nodes in W is at least $\gamma - \rho$.

Let r_1 be the first round in the execution in which the length of the chain of any honest node is at least iL, i.e., all honest nodes are at epoch i.

For any $\epsilon_1 > 0$, let $r_2 = r_1 + (1-\epsilon_1)\frac{L_{\mathtt{reg}}-k}{Q \cdot p}$. From round $r_1 + 1$ to round r_2, nodes make at most $(1-\epsilon_1)\frac{L_{\mathtt{reg}}-k}{p}$ queries to random oracle in which the probability of success in generating a new block of a single query is p. We represent each query by a Bernoulli random variable with an expected value of p. Let n_{b} be the number of blocks that are generated from round $r_1 + 1$ to round r_2. By using the Chernoff bound in Lemma 10 on at most $(1-\epsilon_1)\frac{L_{\mathtt{reg}}-k}{Q \cdot p}$ Bernoulli random variables, we have,

$$\Pr[n_{\mathrm{b}} < k] 1 - e^{-\Omega(\kappa)}.$$

Using the Chernoff bound in Lemma 10, we have,

$$\Pr[n_{\mathrm{b}} < (1-\epsilon_1)(1+\epsilon_1)(L_{\mathtt{reg}} - k)] 1 - e^{-\Omega(\kappa)},$$
$$\Rightarrow \Pr[n_{\mathrm{b}} < (L_{\mathtt{reg}} - k)] 1 - e^{-\Omega(\kappa)}.$$

Let $\bar{h_{\mathrm{c}}}$ be the number of registration blocks that are generated by the honest nodes that are willing to participate as core nodes from round r_1 to r_2. In each round, the honest nodes in W make at least $(\gamma - \rho)\omega$ queries to the random oracle. Thus, from round r_1 to round r_2, the honest nodes in W make at least $(1-\epsilon_1)\frac{L_{\mathtt{reg}}-k}{p}(\gamma-\rho)\omega$ queries to the random oracle. Here, the probability of success in generating a new registration block of a single query is $p' = p\frac{s}{L_{\mathtt{reg}}}$. We represent each query by a Bernoulli random variable with an expected value of p'. We choose $\epsilon_2 > 0$ such that $1 - \epsilon = (1-\epsilon_1)(1-\epsilon_2)$. Using the Chernoff bound in Lemma 10, we have,

$$\Pr[\bar{h_{\mathrm{c}}} > (1-\epsilon)(\gamma-\rho)(1-\frac{1}{L_{\mathtt{reg}}})\omega \cdot s] > 1 - e^{-\Omega(\kappa)}.$$

At round r_2, with a probability of 1 - $e^{-\Omega(\kappa)}$, we have, 1) the length of the longest chain is at most $iL + L_{\mathtt{reg}} - k$; and 2) the number of registration blocks that are generated by honest nodes in W who are willing to participate as core nodes is at least $(1-\epsilon)(\gamma-\rho)(1-\frac{1}{L_{\mathtt{reg}}})\omega$.

From chain quality property in Lemma 11, with probability $1 - e^{-\Omega(\kappa)}$, there exist an honest block from the $(iL + L_{\mathtt{reg}} - \kappa + 1)$-th blocks to the $(iL + L_{\mathtt{reg}})$-th blocks. Thus, if an honest node register as a core node (by propagating a registration block) before the $(L_{\mathtt{reg}} - k)$-th block of the registration period is generated, it will be selected as a core node in the next epoch (i.e., the registration block is included on the blockchain). Thus, we have,

$$\Pr[h_{\mathrm{c}} \geq \bar{h_{\mathrm{c}}}] > 1 - e^{-\Omega(\kappa)},$$
$$\Rightarrow \Pr[h_{\mathrm{c}} \geq \mathsf{q}_{\mathrm{conc}} \cdot s] > 1 - e^{-\Omega(\kappa)}.$$

Following the same proofs and Theorem 3 (except replacing $\mathsf{q}_{\text{M-adap}}$ by $\mathsf{q}_{\mathrm{conc}}$), we can prove the security of the protocol Π_{CSN} in the presence of an adaptive adversary.

□