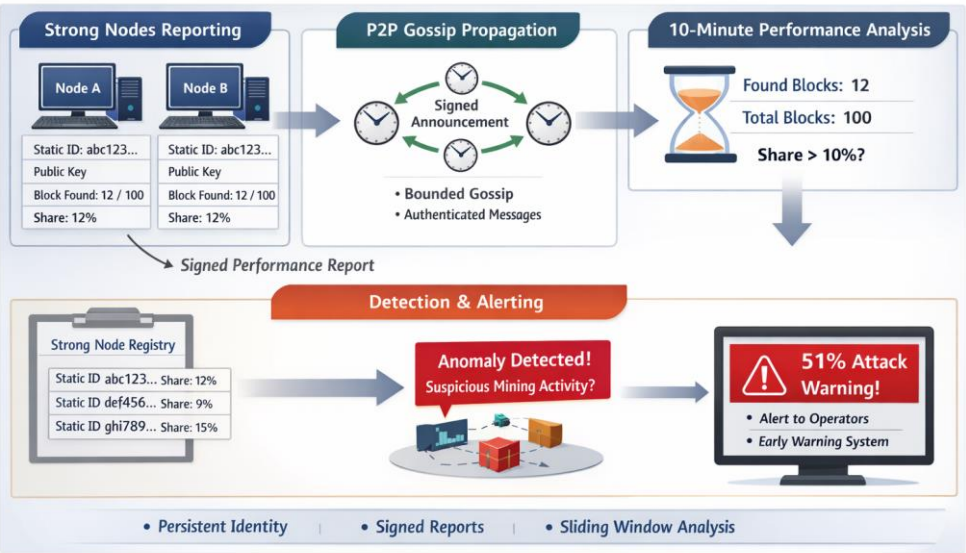
Cryptix (CPAY)

Strong Nodes: Security Architecture

Cryptix Network | https://Cryptix-Network.org | info@cryptix-network.org
Version 2.1

Strong Nodes v2.1: A Overlay System for Early Detection of Potential 51% Attacks



Abstract

This paper presents **Strong Nodes v2.1**, a lightweight, non-consensus overlay mechanism for the **early detection of conditions consistent with a potential 51% attack** in a proof-of-work network. The system introduces a persistent static node identity, signed performance announcements, and a bounded gossip protocol that allows nodes to publish short-term block-

finding activity over a rolling ten-minute observation window. The purpose of the mechanism is not to prove hashrate cryptographically, but to provide an **early warning signal** when one or more nodes appear to accumulate an abnormally large share of recently accepted blocks.

The design explicitly prioritizes operational safety. Strong Nodes v2.1 is optional, locally disableable, resource-bounded, and implemented under a strict safe-fail model: failures in the subsystem never terminate the node process or affect consensus operation. The protocol is hardened against malformed input, replay attempts, and resource exhaustion through validation rules, deduplication, rate limits, bounded persistence, and conservative misbehavior handling.

Strong Nodes v2.1 should therefore be understood as an **early detection and situational awareness system** for possible majority-hashrate concentration events, including the early phases of a potential 51% attack, rather than as a cryptographic proof system.

1. Introduction

A 51% attack is one of the most serious threats to proof-of-work networks. When an attacker, or a coordinated set of actors, acquires a dominant share of effective mining power, the network may be exposed to chain instability, censorship, selfish mining, block suppression, or deep reorganization attacks. In practice, the difficulty is not only mitigation, but **early recognition**. By the time majority control is visible through obvious chain-level effects, the attack may already be underway.

Strong Nodes v2.1 is designed to address this detection gap. It provides a practical mechanism for nodes to observe and exchange signed information about recent block-finding performance. The central hypothesis is simple: if a node repeatedly reports a disproportionately large share of accepted blocks over a short rolling window, this may serve as an **early indicator of abnormal mining concentration**, and therefore as a warning signal for a possible 51% attack scenario.

The system does not attempt to modify consensus, and it does not claim to prove actual hashrate. Instead, it creates a robust and operationally safe monitoring layer that can support node operators, dashboards, and external analysis systems in identifying suspicious concentration patterns before they escalate into a visible majority attack.

2. Problem Statement

Proof-of-work networks generally lack a direct, authenticated, low-overhead mechanism for communicating which participants appear unusually dominant over recent time windows. Existing chain data can show aggregate effects only after they materialize, while peer-to-peer networks typically provide no durable identity or authenticated short-term performance signal for this purpose.

This creates a monitoring blind spot. An attacker preparing a majority-hashrate event may begin to dominate local block observations before the wider network has sufficient evidence to classify the event with confidence. A system intended for early warning should therefore satisfy four requirements.

First, it must provide a persistent identity that allows repeated observations to be linked over time. Second, it must authenticate announcements to prevent trivial spoofing. Third, it must remain bounded and resilient under adversarial network conditions. Fourth, it must fail safely, since any auxiliary detection subsystem must never endanger normal node operation.

Strong Nodes v2.1 is proposed as a direct response to these requirements.

3. System Objective

The objective of Strong Nodes v2.1 is to provide **early detection of potential 51% attack conditions** by surfacing nodes that announce unusually strong recent block-finding performance. The system is designed to detect **signals consistent with emerging mining concentration**, not to establish conclusive proof of malicious control.

Accordingly, Strong Nodes v2.1 is a **discovery and warning overlay**, not a consensus mechanism. It does not influence block validation, fork choice, peer admission, routing fast paths, or mining decisions. Its output is informational and should be interpreted as a warning signal requiring contextual assessment.

4. Threat Model and Detection Model

The system targets scenarios in which a node, pool, or coordinated actor begins to control a substantial share of recently accepted blocks and may be moving toward majority influence over the network. Such behavior may be benign, accidental, or malicious. In all cases, a sudden or sustained short-term dominance pattern is operationally relevant.

Strong Nodes v2.1 assumes that nodes can honestly observe their own accepted block production and their local accepted block universe over a rolling ten-minute window. It further assumes that signed announcements can authenticate the identity that emitted the claim, even though they cannot prove the true external hashrate behind that identity.

The detection model is therefore intentionally modest. The system detects **announced dominance patterns** that are consistent with the early stages of a potential 51% attack. It does not prove actual miner ownership, pool composition, hidden coordination, or real global hashrate. Its value lies in **early warning under uncertainty**.

5. Architecture Overview

Strong Nodes v2.1 introduces a persistent Strong-Node identity, stored outside the chain database under the node’s application data directory. This identity is derived from a dedicated local keypair and remains stable across restarts. The associated public key defines a static identifier, while the secret key is used to sign Strong-Node announcements.

Each qualified node periodically emits a signed announcement containing its static identity, public key, recent ten-minute performance metrics, a sequence number, a time window, and an optional self-reported IP address claim. These announcements are propagated using a capability-gated gossip mechanism restricted to peers advertising Strong-Nodes support.

Remote announcements are accepted only if they pass strict structural validation, network matching, time-window checks, signature verification, identifier-to-public-key consistency

rules, replay protection, and resource limits. Accepted entries are maintained in a bounded local registry and exposed through a dedicated RPC endpoint for operator visibility and downstream monitoring.

The architecture is intentionally isolated from consensus. The Strong-Nodes subsystem operates as an optional runtime module that can be disabled by configuration or deactivated automatically upon internal failure, without affecting node correctness or liveness.

6. Static Identity and Persistence

A central design requirement is that repeated announcements must be attributable to a stable node identity. Strong Nodes v2.1 therefore defines a dedicated persistent identity separate from any preexisting transient or protocol-level node identifier. The identity is generated once, stored locally, and reused across restarts.

Persistence is implemented outside the chain database under a dedicated directory. Two files are maintained: one for the node identity and one for the bounded local registry of observed Strong-Nodes entries. File handling is hardened through strict size limits, schema checks, field-length checks, quarantine of corrupt files, atomic write procedures based on temporary files and rename operations, and restricted file permissions for sensitive key material.

This persistence model serves two purposes. It preserves identity continuity for meaningful longitudinal observation, and it ensures that the subsystem remains operationally isolated from consensus-critical storage.

7. Qualification Logic for Early Warning

The system uses a rolling ten-minute observation window derived from locally accepted block events. Within this window, `found_blocks_10m` counts successfully accepted locally submitted blocks, while `total_blocks_10m` counts all locally accepted submitted and relayed blocks.

A node becomes eligible for announcement only if three quantitative conditions hold simultaneously: its observed share exceeds ten percent, the total observed block count is at least sixty, and the node has found at least three blocks within the window. To suppress statistical noise and avoid oscillation around the threshold, qualification requires the criteria to be satisfied across two consecutive evaluations. Dequalification likewise requires two consecutive evaluations below the lower hysteresis threshold or below the minimum-sample conditions.

This design is intended to reduce false positives from small samples while still allowing the network to surface unusually strong participants early enough to support operator awareness. The threshold is not a proof of attack. Rather, it is a signal that a node appears unexpectedly dominant over the recent observation horizon and may merit attention as part of a possible 51% attack buildup.

8. Announcement Protocol

Strong-Nodes announcements are transported as a dedicated peer-to-peer payload protected by a new service capability bit. The message includes schema version, network identifier,

static identity, x-only public key, monotonic sequence number, observation window bounds, found and total block counts, send timestamp, optional claimed IP address, and a Schnorr signature.

The signed preimage is deterministically serialized and domain-separated so that signatures cannot be replayed across unrelated message contexts. The sequence number provides strict ordering for each identity. For a given static identity, only announcements with strictly larger sequence numbers are accepted. Equal or older sequence numbers are discarded. Additional time-based plausibility checks reject announcements that are excessively old or implausibly far in the future.

Announcements are forwarded only in bounded fanout form and only to peers that explicitly advertise support for the feature. This ensures backward compatibility with legacy nodes and avoids unnecessary propagation overhead.

9. Security Properties

The protocol is designed to authenticate identity and preserve message integrity, while remaining honest about what it cannot prove. A valid Strong-Nodes announcement proves that the holder of the corresponding private key signed a claim about recent performance. It does not prove that the reported performance corresponds to actual global hashrate, nor does it prove that one static identity corresponds to exactly one miner, pool, or machine.

Within that boundary, the design provides useful security properties. It prevents trivial spoofing of another node's identity. It prevents unauthenticated mutation of announced metrics. It constrains replay through monotonic sequence numbers and timestamp checks. It reduces abuse through per-peer rate limits, caps on new identities, deduplication caches, hard message-size limits, bounded registries, and deterministic eviction rules.

This security model is sufficient for an **early warning system**. It is not sufficient for adversarially strong attribution or formal proof of majority control, and the paper intentionally does not claim otherwise.

10. Safe-Fail Operational Model

A key principle of Strong Nodes v2.1 is that the node must continue operating correctly even if the Strong-Nodes subsystem fails completely. For that reason, the subsystem is implemented as an optional runtime component whose activation depends on configuration, hardfork state, and runtime availability.

If initialization fails because of file corruption, key handling problems, or internal state issues, the subsystem is disabled locally and the node continues running. If repeated internal subsystem errors occur during runtime, a circuit breaker disables the feature for the remainder of the process lifetime. Importantly, malformed or malicious remote input does not count toward this circuit breaker. Adversarial payloads are handled through dropping, optional strike accumulation, and existing ban logic, rather than by allowing remote actors to disable the feature indirectly.

This safe-fail model ensures that Strong Nodes v2.1 can be deployed in production as a monitoring overlay without creating a new availability risk for the core node.

11. Observability and Operator Use

The subsystem exposes its local view through a dedicated RPC method, `GetStrongNodes`. The response includes activation status, runtime state, disabled-state observability fields, and the current set of locally known announced Strong-Nodes entries. Each entry reports the static identifier, public key, announced and observed metadata, source classification, stale-state information, and the explicit field `performance_verified = false`.

This interface is intended for monitoring systems, operator dashboards, alerting pipelines, and forensic tooling. Its purpose is to make potential concentration events visible at an early stage. For example, if one or several identities begin to show unusually high recent share values across multiple observation windows, operators may interpret that as an early signal of a possible 51% attack trajectory or coordinated mining centralization event.

12. Limitations

Strong Nodes v2.1 is not a proof of hashrate and should not be presented as one. It relies on signed self-reports built from local observations. As a result, different nodes may observe different local block universes, and an attacker may attempt to game perception by shaping visibility or deploying multiple identities.

The system also does not attempt to solve identity linking, key continuity after loss, or attribution across entities. Key rotation creates a new identity. Claimed IP addresses are self-reported and therefore informational only, while transport-observed sender addresses may be affected by NAT, relays, or infrastructure topology.

These limitations do not nullify the usefulness of the mechanism. They define its correct role: a practical, bounded, signed, and operationally safe **early detection layer** for conditions that may be consistent with an emerging 51% attack.

13. Conclusion

Strong Nodes v2.1 provides a pragmatic approach to one of the hardest operational problems in proof-of-work systems: recognizing suspicious concentration of block-finding power early enough to matter. By combining persistent node identity, signed short-horizon performance announcements, bounded gossip propagation, strict validation, and a fail-open runtime model, the system offers a realistic path toward **early warning of potential 51% attack conditions** without modifying consensus.

Its contribution is not cryptographic certainty, but actionable visibility. In networks where majority attacks are most dangerous precisely because they are recognized too late, such visibility can be valuable. Strong Nodes v2.1 should therefore be understood as a safe, low-overhead, non-consensus mechanism for early detection of mining dominance patterns that may precede or accompany a 51% attack. Important: This system is a monitoring/early warning system, not an evidence or defense mechanism.