Luvion Technical Summary Luvion Technical Summary

Scope

Scope

This document summarizes the repository implementation of the Luvion 22-of-33 threshold signature path.

The implementation is a two-round threshold signing system over the ML-DSA-65 parameter set. It supports:

33 total signer nodes.

22 required signer nodes per signing committee.

Shamir-shared secret material for

$s1$

,

$s2$

, and

$t0$

.

Persistent TCP signer nodes.

A

cluster_sign

coordinator that probes nodes, selects a committee, runs the signing ceremony, aggregates partial responses, and verifies successful

signatures.

The current

gen_shares

path is a synthetic dealer for test and demo material. It is not production DKG.

Key Files

Key Files

File

File

Responsibility

Responsibility

src/params.rs

ML-DSA-65 constants and Luvion threshold constants.

src/signing.rs

Core 22-of-33 signing math: round 1, challenge, round 2, aggregation,

verification.

src/shamir.rs

Shamir sharing and reconstruction helpers for polynomial vectors.

src/share_io.rs

On-disk public key and node share formats.
src/wire.rs
Versioned frame protocol and DTOs shared by coordinator and nodes.
src/bin/gen_shares.rs
Synthetic dealer that creates
pk.bin
and 33 node share files.
src/bin/node.rs
Persistent signer node process.
src/bin/cluster_sign.rs
TCP cluster coordinator and CLI.
src/bin/_cluster/mod.rs
Shared cluster driver logic for probing and ceremony execution.
tests/cluster_sign_tcp.rs
Persistent TCP cluster integration tests.
tests/coordinator_integration.rs
Stdio coordinator integration tests.
tests/ceremony_22_of_33.rs
Core 22-of-33 ceremony coverage.
Parameters
Parameters
The repository uses ML-DSA-65 / Dilithium3-style dimensions plus Luvion-specific threshold parameters.
Parameter
Parameter
Value
Value
Meaning
Meaning
N_PARTIES
33
Total signing nodes.
THRESHOLD
22
Minimum nodes required for one signature
attempt.
N
256

Polynomial degree.

Q
8,380,417
Ring modulus.
K
6
Public matrix rows.
L
5
Public matrix columns.
ETA
4
Secret coefficient bound.
GAMMA1
524,288
Mask bound base.
GAMMA2
261,888
Low/high bits decomposition bound.
TAU
49
Challenge polynomial weight.
BETA
196
TAU * ETA
.
OMEGA
55
Maximum hint weight.
D
13
Public-key dropped bits.
Y_BOUND_PER_NODE
23813
Per-node mask bound for threshold aggregation.
Parameter
Parameter
Value

Value

Meaning
Meaning
The per-node mask bound is chosen so the aggregate response can satisfy the ML-DSA norm requirement:
THRESHOLD * Y_BOUND_PER_NODE + TAU * ETA < GAMMA1 - BETA
Key Material Model
Key Material Model
The signing key is represented as ML-DSA material split across nodes.
The synthetic dealer in
gen_shares
derives:
1
.
pk_seed
.
2
.
Full secret vector
s1
.
3
.
Full secret vector
s2
.
4
.
Public matrix
A = ExpandA(pk_seed)
.
5
.
Public value:
t = A * s1 + s2
6
.
Power2Round decomposition:

(t1, t0) = Power2Round(t)
7

.
Public key:
pk = (pk_seed, t1)
8
.
Shamir shares for each node:
s1_i = ShamirShare(s1, i)
s2_i = ShamirShare(s2, i)
t0_i = ShamirShare(t0, i)
Each node receives one
NodeShare
containing:
index
dealer_fingerprint
flags
s1 share
s2 share
t0 share
The share files are marked with
SHARE_FLAG_SYNTHETIC_DEALER
. That flag is a clear signal that the material came from the synthetic dealer and is not production key material.
On-Disk Formats
On-Disk Formats
Public key files use:
magic: LVPK
version: 1
seed: 32 bytes
t1: K * N * 4 bytes
Node share files use:
magic: LVNS
version: 1
index: u64
dealer_fingerprint: u64
flags: u8
s1: PolyVecL

s2: PolyVecK
t0: PolyVecK

When a node starts, it loads
pk.bin
and its own
node<N>.bin
, verifies that the node id matches the share index, and verifies that the share dealer fingerprint matches the public key.
Network Runtime
Network Runtime
The networked signing path has two roles:
Role
Role
Binary
Binary
Responsibility
Responsibility
Signer node
node
Holds one share, listens on TCP, responds to protocol frames.
Coordinator
cluster_sign
Connects to nodes, chooses 22 nodes, runs the signing attempt, aggregates, verifies.
The node process can run over stdio or TCP. The persistent cluster path uses TCP.
The coordinator supports:
Endpoint discovery from
--base-port
or explicit
--endpoints
.
Parallel or sequential probing.
Committee strategies:
first-alive-22
,
lowest-rtt-22
, and
random

.
t0
 fetch modes:

on-demand
 and
prefetch
.
Reconnect handling for transient transport failures.
JSON timing logs for each attempt and the full run.
Wire Protocol
Wire Protocol
The wire layer is versioned and length-prefixed.
Frame structure:
u32 length
u8 wire_version
u8 msg_kind
... bincode body
Frame constraints:
Field
Field
Constraint
Constraint
WIRE_VERSION
Must match the local version.
MAX_FRAME_LEN
1 << 20
 bytes.
msg_kind
Must be one of the known protocol message
kinds.
Message kinds:
Kind
Kind
Direction
Direction
Purpose
Purpose
Ping

Coordinator -> node
Liveness probe.
Pong

Node -> coordinator
Node id and binary version.
Round1Request
Coordinator -> node
Ask node to sample mask and
commit.
Round1Response
Node -> coordinator
Return
W_i
and commitment hash.
Round2Request
Coordinator -> node
Send challenge and participant set.
Round2Response
Node -> coordinator
Return partial
z_i
and
$cs2_i$
.
GetT0Request
Coordinator -> node
Fetch the node's
t0
share.
GetT0Response
Node -> coordinator
Return
$t0_i$
.
Error
Node -> coordinator
Protocol-level error response.
Shutdown

Coordinator -> node
Shutdown control message.
Kind
Kind

Direction
Direction
Purpose
Purpose
Signing Protocol
Signing Protocol
Committee Selection
Committee Selection
cluster_sign
first probes all configured endpoints and keeps live connections. If fewer than 22 nodes are alive, it exits with the quorum failure code.
After probing, it selects exactly 22 nodes using the configured strategy:
first-alive-22
: lowest sorted live node ids.
lowest-rtt-22
: lowest measured probe RTT, tie-broken by node id.
random
: random 22 from the live set.
The selected participant list is sorted and must contain unique ids.
Attempt Setup
Attempt Setup
For each signing attempt, the coordinator creates:
request_id: 32 random bytes
seed_i: 64 random/domain-separated bytes per participant
nonce_i: 32 random/domain-separated bytes per participant
The
request_id
binds round 1 and round 2 on each node. Nodes cache round-1 mask state by
request_id
and consume it exactly once during round 2.
Round 1: Commitment
Round 1: Commitment
For participant
i

, the node:
1
.
Validates the
request_id

 is not already active in its cache.
2
.
Samples a fresh masking vector:
y_i <- bounded sample with |coeff| < Y_BOUND_PER_NODE
3
.
Computes:
W_i = A * y_i
4
.
Computes a commitment hash:
commit_i = H("luvion_round1_commit_v1", i, nonce_i, W_i)
5
.
Stores
y_i
,
nonce_i
, and
commit_i
 in the node cache under
request_id
.
6
.
Returns
W_i
 and
commit_i
.
Round1Output
 redacts

y_i
from debug output and zeroizes
y_i
on drop.
Challenge
Challenge

The coordinator aggregates:
W = sum(W_i)
Then it computes:
W1 = HighBits(W)
mu = SHAKE256("luvion_challenge_mu_v1", pk.seed, pk.t1)
seed = SHAKE256("luvion_challenge_seed_v1", mu, W1, message)
c = poly_challenge(seed)
c
is the sparse challenge polynomial with ML-DSA challenge weight.
Round 2: Partial Response
Round 2: Partial Response
For participant
i
, the node:
1
.
Looks up and removes its cached
y_i
using
request_id
.
2
.
Validates the participant set has at least 22 sorted unique entries and contains itself.
3
.
Computes the Lagrange coefficient:
lambda_i = LagrangeCoefficient(i, participants)
4
.
Computes its partial response:
z_i = y_i + lambda_i * c * s1_i

cs2_i = lambda_i * c * s2_i
5
.
Returns
z_i
 and

cs2_i
.
Only the long-term secret shares are Lagrange-scaled. The fresh mask
y_i
 is not Lagrange-scaled; the aggregate mask is the direct sum of the per-node masks.
t0
t0
 Fetch
 Fetch
The aggregator also needs the threshold reconstruction of
t0
 to compute the ML-DSA hint.
cluster_sign
 can fetch
t0
 in two modes:
Mode
Mode
Behavior
Behavior
on-demand
Fetch
t0_i
 inside the attempt.
prefetch
Fetch
t0_i
 before attempts and reuse the
cache.
The optimized TCP path defaults to
prefetch
 because it removes

GetT0
latency from the signing attempt itself.
Aggregation
Aggregation
The coordinator aggregates round-2 outputs:
z = sum(z_i) = sum(y_i) + c * s1
cs2 = sum(cs2_i) = c * s2

It reconstructs
t0
from shares:
t0 = sum(lambda_i * t0_i)
ct0 = c * t0
Then it checks the ML-DSA rejection conditions:
||z||_inf < GAMMA1 - BETA
||LowBits(W - cs2)||_inf < GAMMA2 - BETA
||ct0||_inf < GAMMA2
hint_weight <= OMEGA
If any check fails, the attempt is classified as a probabilistic rejection. This is not a transport failure.
Why
Why
r0_norm
r0_norm
Rejections Are Expected
Rejections Are Expected
r0_norm
is the infinity norm of the low bits in:
r0 = LowBits(W - c*s2)
The acceptance rule is:
r0_norm < GAMMA2 - BETA
With the current ML-DSA-65 parameter set, this means:
GAMMA2 = 261888
BETA = 196
limit = 261692
This check exists to keep the final threshold signature distribution compatible with ML-DSA rejection sampling. If a candidate attempt has
r0_norm
above the limit,
the coordinator discards that attempt and retries with fresh per-node randomness. This protects against

secret-dependent leakage and ensures the verifier can
recover the correct high bits from the final hint.
An
r0_norm
 rejection means the network path and 22-node ceremony completed, but the sampled candidate did not pass the ML-DSA acceptance bounds. It is
normal for some attempts to be rejected before a later attempt returns

verified_signature
. It only becomes a failed signing run if every attempt is rejected until
-
-max-attempts
 is exhausted, which returns exit
4
.
If all checks pass, the final signature is:
signature = (c, z, hint)
where:
hint = MakeHint(-ct0, W - cs2 + ct0)
Verification
Verification
Verification checks:
1
.
||z||_inf < GAMMA1 - BETA
.
2
.
hint_weight <= OMEGA
.
3
.
Recompute:
Az = A * z
ct1 = c * (t1 << D)
W' = Az - ct1
W1' = UseHint(W', hint)
c' = Challenge(pk, W1', message)
4

.
Accept only if:
c' == c
cluster_sign
 verifies every successfully aggregated signature before returning success.
Attempt Outcomes and Exit Codes
Attempt Outcomes and Exit Codes
cluster_sign

 distinguishes protocol rejection from transport failure.
Exit
Exit
Code
Code
Meaning
Meaning
0
Verified signature produced.
3
Fewer than 22 alive nodes.
4
Max attempts exhausted due to probabilistic rejection.
5
Transport failure after retry budget.
1
CLI, configuration, decode, key loading, or internal
error.
Exit
Exit
Code
Code
Meaning
Meaning
For this repository, exit
4
 means the selected 22 nodes completed full ceremony attempts without transport failure, but no candidate signature passed the ML-DSA
rejection checks within the configured attempt budget. Demo scripts use a larger attempt budget and require exit

0
.
The most common retry reason in WAN demos is usually
reject_reason=r0_norm
. That status is expected protocol behavior, not a node failure. A successful run
may still contain earlier rejected attempts, as long as a later attempt returns
classification=verified_signature
and exit

0
.
Node State and Replay Protection
Node State and Replay Protection
Each node has a
YntCache
keyed by
request_id
.
The cache stores:
y_i
nonce_i
commitment_hash
created_at
Properties:
Round 1 rejects duplicate active
request_id
s.
Round 2 consumes and removes the cached entry.
Unknown or expired
request_id
s are rejected.
Cache entries expire by TTL.
Cache size is bounded.
Secret mask
y_i
is zeroized when dropped.
This prevents a coordinator from reusing a node's one-time mask across different challenges.
Transport Robustness
Transport Robustness

The TCP path includes:
Per-connection read and write timeouts.
Parallel probing with probe RTT measurement.
Validation that probed node id matches the expected node id.
Version validation on wire frames.
Persistent committee connections reused across attempts.
Reconnect logic on transport errors.
JSON logs for attempts and run summaries.
The node TCP listener accepts concurrent connections so one idle client cannot block the signer process

from serving valid coordinators.
Performance Characteristics
Performance Characteristics
The measured optimized local path is approximately:
Metric
Metric
Local Persistent
Local Persistent
Cluster
Cluster
Probe
36ms
t0
prefetch
41ms
Ceremony attempt
130ms
Wall time
233ms
Reconnects
0
Transport failures
0
The measured optimized DigitalOcean 25-droplet path is approximately:
Metric
Metric
DigitalOcean Spread-25
DigitalOcean Spread-25
Probe

260ms
t0
 prefetch
195ms
Ceremony attempt
475ms
Full
cluster_sign
 invocation

1.128s
Reconnects
0
Transport failures
0
Metric
Metric
DigitalOcean Spread-25
DigitalOcean Spread-25
The main latency contributors in WAN mode are network probe RTTs,
t0
 prefetch, and cross-region round-trip time during round 1 and round 2.
Current Limitations
Current Limitations
The repository currently uses a synthetic dealer for easy testing. Production deployment requires a real DKG or secure key-generation ceremony.
The signing math uses schoolbook polynomial multiplication in correctness-critical signing paths. Comments in the code note an unresolved NTT/Montgomery-
convention issue, so the NTT fast path is not used for those operations.
Very small attempt budgets can still produce exit
4
 because ML-DSA rejection sampling may reject candidate attempts. Use the demo defaults or increase --max-
attempts
 when a verified signature is mandatory.
The current demo and test deployment model exposes raw TCP node ports. A production system should add authenticated transport, access control, key
management, observability, and stronger operator controls.
Validation Coverage

Validation Coverage

Important validation paths include:

Command

Command

Coverage

Coverage

cargo check

Type and compile validation.

cargo test --test ceremony_22_of_33

Core threshold ceremony behavior.

cargo test --test coordinator_integration

Stdio coordinator integration.

cargo test --test cluster_sign_tcp -- --test-threads=1

Persistent TCP cluster behavior, retries, timeout handling, idle connection handling, probe timeout behavior.

cargo build --release --bin gen_shares --bin node --bin cluster_sign

Release binaries used by local and real-world runs.

Recent validation confirmed:

TCP idle connections do not block signer nodes.

Probe ping uses the configured I/O timeout.

Wire frame version mismatches are rejected.

The persistent local 33-node flow can run a 22-node sign with zero reconnects and zero transport failures.

The DigitalOcean 25-droplet flow can deploy 33 nodes, sign with 22 endpoints, and clean up all droplets.