Boing Network — Testnet

> ***Purpose:*** *Single guide for joining the testnet, using the Testnet Portal, and the incentivized testnet program (readiness, promotion, mainnet migration).*
> ***References:*** *TESTNET-RPC-INFRA.md (single map: testnet ops + public RPC + infra), RUNBOOK.md, READINESS.md, RPC-API-SPEC.md, INFRASTRUCTURE-SETUP.md, VIBEMINER-INTEGRATION.md*

Table of Contents

Part 1 — Join Testnet

Run nodes on the testnet, get testnet BOING from the faucet, and join as a validator or developer.

1. Single node vs multi-node

Mode	Use case	How to run
Single node	Local dev, trying the chain alone, no P2P	Run `boing-node` **without** `--p2p-listen`. The node runs in isolation: it produces blocks if `--validator`, and serves RPC. No other peers.
Multi-node testnet	Public testnet: many nodes syncing and validating together	Run `boing-node` **with** `--p2p-listen` and `--bootnodes`. Your node joins the P2P network, syncs blocks from peers, and (if `--validator`) can produce blocks when it's the leader.

Summary:

- **Single node** = one machine, one chain, no peer discovery. Good for "run a chain on my laptop."
- **Multi-node** = many nodes connected via P2P; they discover each other using **bootnodes** and stay in sync.

2. What are bootnodes?

Bootnodes are well-known peer addresses that new nodes **dial on startup** to join the network. Without them, a node with P2P enabled would only see peers on the same LAN (via mDNS). With bootnodes, your node can connect to the public testnet even from home.

- **Format:** Multiaddr, e.g. `/ip4/1.2.3.4/tcp/4001` (IP + port where a testnet node is listening).
- **Who runs them:** Usually the team or community; they run a node with a stable IP and publish its address.
- **How you use them:** Pass `--bootnodes /ip4/.../tcp/4001,/ip4/.../tcp/4002` when starting your node (comma-separated).

Example (replace with real testnet bootnodes):

```
./boing-node --p2p-listen /ip4/0.0.0.0/tcp/4001 \
  --bootnodes /ip4/testnet.boing.network/tcp/4001 \
  --validator --rpc-port 8545 --data-dir ./data
```

3. Running a single node (no P2P)

```
cargo build --release

# Full node (no block production)
./target/release/boing-node --rpc-port 8545 --data-dir ./data

# Validator (produces blocks)
./target/release/boing-node --validator --rpc-port 8545 --data-dir ./data
```

RPC: `http://127.0.0.1:8545/` . No bootnodes needed.

4. Running on the multi-node testnet

1. **Build**
   `cargo build --release`
2. **Start with P2P + bootnodes** (use the bootnode list from Join Testnet or §6 below):

   ```
   ./target/release/boing-node \
     --p2p-listen /ip4/0.0.0.0/tcp/4001 \
     --bootnodes "<BOOTNODE_1>,<BOOTNODE_2>" \
     --validator \
     --rpc-port 8545 \
     --data-dir ./boing-data
   ```
3. **Get testnet BOING** from the Faucet (see §5).
4. **Stake** by submitting a `Bond` transaction via RPC so you can participate as a validator (validator set is derived from top stakers).

5. Faucet (testnet BOING)

Testnet nodes can expose a **faucet** so users get test BOING without mining.

5.1 RPC method: `boing_faucetRequest`

When the node is started with `--faucet-enable` , it accepts:

Method	Params	Description
`boing_faucetRequest`	`[hex_account_id]`	Send **1,000** testnet BOING to the given account (32-byte hex). Rate limit: **1 request per 60 seconds per account**.

Example (curl):

```
# Your account ID as 32-byte hex (e.g. from your wallet)
curl -s -X POST http://127.0.0.1:8545/ -H "Content-Type: application/json" \
```

```
  -d '{"jsonrpc":"2.0","id":1,"method":"boing_faucetRequest","params":
["0xYOUR_32_BYTE_ACCOUNT_ID_HEX"]}'
```

Response (success):

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "result": {
    "ok": true,
    "amount": 1000,
    "to": "hex_account_id",
    "message": "Check your wallet; tx is in the mempool."
  }
}
```

Notes:

- Only nodes started with `--faucet-enable` support this. **Do not use on mainnet.**
- The faucet account is funded at genesis with 10,000,000 testnet BOING; each request sends 1,000.
- If you get "Faucet cooldown", wait 60 seconds and try again for the same account.

5.2 Public faucet page

The website provides a **dedicated faucet page** at boing.network/faucet (or your deployment path). Enter your account ID (hex) and request testnet BOING; the page calls the testnet RPC's `boing_faucetRequest` for you.

Testnet RPC URL: `https://testnet-rpc.boing.network/` (also on the Testnet page).

5.3 Native constant-product AMM pool (chain 6913)

The in-ledger **MVP constant-product pool** is a normal **32-byte contract** `AccountId`. Integrators set it in dApp config — on **boing.finance**: `frontend/src/config/boingCanonicalTestnetPool.js` (constant `CANONICAL_BOING_TESTNET_NATIVE_CP_POOL_HEX`) and/or `REACT_APP_BOING_NATIVE_AMM_POOL` for Pages builds, plus `nativeConstantProductPool` in `contracts.js` for chain **6913**. The **canonical public testnet** hex is the table below and in RPC-API-SPEC.md § Native constant-product AMM; optional TypeScript mirror: `CANONICAL_BOING_TESTNET_NATIVE_CP_POOL_HEX` in `boing-sdk`. New pool rotations: OPS-CANONICAL-TESTNET-NATIVE-AMM-POOL.md.

Canonical public testnet pool `AccountId`	`0xffaa1290614441902ba813bf3bd8bf057624e0bd4f16160a9d32cd65d3f4d0c2` — same as RPC-API-SPEC.md § Native constant-product AMM; **v1** + CREATE2 (`NATIVE_CP_POOL_CREATE2_SALT_V1`). Also on the website Join Testnet page (`website/src/config/testnet.ts`).

Storage keys and calldata: NATIVE-AMM-CALLDATA.md. SDK examples: `BOING_POOL_HEX` in examples/native-boing-tutorial.

6. Bootnode list (testnet)

When the testnet is live, the canonical list will be kept at:

- **Website:** boing.network/testnet/join and Bootnodes (driven by `website/src/config/testnet.ts` or env `PUBLIC_BOOTNODES`)

- **This repo:** Below (update before testnet launch)
- **Infrastructure setup:** INFRASTRUCTURE-SETUP.md

Bootnode	Multiaddr	Notes
Primary	`/ip4/73.84.106.121/tcp/4001`	Faucet + RPC via testnet-rpc.boing.network (Cloudflare tunnel)
Secondary	`/ip4/73.84.106.121/tcp/4001`	Run via `scripts/start-bootnode-2`

Launch checklist (to open testnet):

1. **Bootnodes:** Run at least 2 nodes with stable IPs and `--p2p-listen /ip4/0.0.0.0/tcp/4001`. Add their multiaddrs to the table above and to `website/src/config/testnet.ts` (or set `PUBLIC_BOOTNODES` when building the website).
2. **Public RPC:** Run a node with `--faucet-enable` behind a public URL (e.g. `https://testnet-rpc.boing.network/`). Set `PUBLIC_TESTNET_RPC_URL` when building the website so the faucet page defaults to it.
3. **Genesis:** All nodes must use the same genesis so the faucet account has 10M testnet BOING.
4. **Docs:** For the full pre-launch checklist and incentive program see Part 3 — Incentivized Testnet below. For the critical path (bootnodes → RPC → VibeMiner / boing.observer), see READINESS.md §3.

Until then, you can run a multi-node testnet locally by starting two nodes and having the second dial the first:

Terminal 1 (first node):

```
./target/release/boing-node --p2p-listen /ip4/127.0.0.1/tcp/4001 --validator --rpc-port 8545
```

Terminal 2 (second node, dials the first):

```
./target/release/boing-node --p2p-listen /ip4/127.0.0.1/tcp/4002 \
  --bootnodes /ip4/127.0.0.1/tcp/4001 \
  --rpc-port 8546
```

7. One-click mining / validator UI (VibeMiner)

For users who prefer a **desktop UI** instead of the terminal, Boing testnet can be used with **VibeMiner**, which provides one-click mining/validating. See VIBEMINER-INTEGRATION.md for how networks (including Boing) integrate and how to run a node via VibeMiner.

8. Incentivized testnet (summary)

When the Boing team runs an **incentivized testnet** (rewarding validators, developers, and users), the same testnet setup applies: use the published bootnodes and public RPC, get testnet BOING from the faucet, and stake to validate. For incentive rules, duration (e.g. 2–4 weeks), launch checklist, promotion, and mainnet migration, see Part 3 — Incentivized Testnet below. Check the website and announcements for the current phase and any leaderboards or reward criteria.

9. Ship a new testnet node binary (release tags)

GitHub Actions builds **`release-{linux-x86_64,macos-aarch64,windows-x86_64}.zip`** when you push a tag matching `testnet*` or `v*` (see `.github/workflows/release.yml`).

1. Ensure `main` has the code you want shipped.
2. Create and push an **annotated** tag (example `testnet-v0.1.7`):

```
git checkout main
git pull
git tag -a testnet-v0.1.7 -m "Testnet node: describe changes"
git push origin testnet-v0.1.7
```

3. Wait for workflow **Release binaries** to finish. For `testnet*` tags the workflow **publishes** the release immediately so `https://github.com/.../releases/download/<tag>/...` works. For `v*` tags it may create a **draft** until you click **Publish release** in GitHub → Releases.
4. If downloads return HTTP **404**, the release is usually still a **draft** or uploads are incomplete — open **Releases** and confirm the assets exist.
5. Refresh Boing website D1 listing SHAs (from repo root or `website/`):

```
node scripts/network-listings-release-sql.mjs testnet-v0.1.7
node scripts/network-listings-release-sql.mjs testnet-v0.1.7 --apply
# or: cd website && node scripts/network-listings-release-sql.mjs testnet-v0.1.7 [--
apply]
```

6. Bump `BOING_TESTNET_DOWNLOAD_TAG` in `website/functions/api/networks.js` (and `BOING_ZIP_SHA` if you use URL rewrite helpers) so `GET /api/networks` returns the new `meta.boing_testnet_download_tag`. Deploy **boing.network**. Then align **VibeMiner** defaults with `meta` — VIBEMINER-INTEGRATION.md §6.

Also: WEBSITE-AND-DEPLOYMENT.md (D1 / listings), PUBLIC-RPC-NODE-UPGRADE-CHECKLIST.md before pointing users at a new RPC surface.

Part 2 — Testnet Portal

A dedicated testnet portal where participants register as **developer**, **user**, or **node operator**, with role-specific dashboards, community pages, and metrics. Replaces the standalone quests page with a unified experience. References: DEVELOPMENT-AND-ENHANCEMENTS.md, BOING-BLOCKCHAIN-DESIGN-PLAN.md.

2.1 Overview

The **Testnet Portal** is the single place to:

- **Sign up** — Register with a testnet account ID and choose role: **Developer**, **User**, or **Node operator**.
- **Track metrics** — Each role has a dedicated dashboard and community page with relevant metrics.
- **Qualify for rewards** — Registration and activity feed into the incentivized testnet rules (Community & Grants pool). Developers have a dedicated portion of the ecosystem pool for **successful dApps** (mainnet); testnet dev participation qualifies for grants and recognition.

Site structure:

Path	Purpose

`/testnet`	Portal landing: choose role, register, or go to dashboard
`/testnet/register`	Registration form (account ID + role + optional contact)
`/testnet/developers`	Developers community + dashboard (dApps, metrics, link to dApp incentive pool)
`/testnet/users`	Users community + dashboard (quests, faucet, feedback)
`/testnet/operators`	Node operators community + dashboard (leaderboard, uptime, blocks)

The **Join Testnet** hub is at `/testnet/join` (bootnodes, faucet, single-vs-multi). It links prominently to the portal for registration and dashboards. **Community Quests** live under **Users** at `/testnet/users/quests` (the old `/network/quests` URL redirects there).

2.2 Registration

- **Required:** Account ID (32-byte hex), Role (developer | user | node_operator).
- **Optional:** Email, Discord handle, GitHub username (for devs), node multiaddr (for operators).
- **Storage:** D1 table `portal_registrations`. One registration per account ID; role can be updated (e.g. user → developer) until testnet end.
- **API:** `POST /api/portal/register` — validate account_id and role; insert or update; return success. Rate limit per IP.
- **Verification:** No email verification required for Phase 1. Optional: later, link registration to on-chain identity (e.g. sign a message with the account key).

2.3 Role-Specific Dashboards & Metrics

Developers

- **Community page:** `/testnet/developers` — Who's building, links to docs, SDK, CLI, and **success-based dApp incentives**.
- **Dashboard (per account):** Registered dApps; metrics placeholder (fees, tx count when indexer/RPC supports it). **Rewards:** Testnet BOING, bug bounties, dApp recognition; mainnet eligibility for grants and dApp incentive pool.

Users

- **Community page:** `/testnet/users` — Quests, faucet link, feedback. **Dashboard:** Quest progress, faucet/first tx badges. **Rewards:** Faucet + quests + feedback; optional capped NFT/mainnet recognition per Incentivized Testnet Rules.

Node operators

- **Community page:** `/testnet/operators` — How to run a node, bootnodes, VibeMiner, leaderboard. **Dashboard:** Validator metrics (blocks, uptime when available), leaderboard. **Rewards:** Testnet BOING (blocks) + leaderboard/uptime; mainnet allocation or NFT for top N (capped).

2.4 Data Model (D1)

```
CREATE TABLE IF NOT EXISTS portal_registrations (
  account_id_hex TEXT PRIMARY KEY,
  role TEXT NOT NULL,
  email TEXT,
```

```
  discord_handle TEXT,
  github_username TEXT,
  node_multiaddr TEXT,
  created_at TEXT NOT NULL,
  updated_at TEXT NOT NULL
);

CREATE TABLE IF NOT EXISTS portal_dapps (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  contract_hex TEXT NOT NULL,
  owner_account_hex TEXT NOT NULL,
  name TEXT,
  registered_at TEXT NOT NULL,
  UNIQUE(contract_hex)
);
CREATE INDEX IF NOT EXISTS idx_portal_dapps_owner ON portal_dapps(owner_account_hex);
```

2.5 API (Cloudflare Worker)

Method	Path	Purpose
POST	`/api/portal/register`	Register or update role
GET	`/api/portal/me?account_id_hex=0x...`	Registration + role-specific summary
GET	`/api/portal/leaderboard?role=node_operator&limit=20`	Leaderboard for operators
POST	`/api/portal/dapps`	Register a dApp (dev only)
GET	`/api/quests/status?account_id_hex=0x...`	Quest progress for users

2.6 Implementation Phases

Phase	Scope
Phase 0	Static portal: landing, register, community pages with placeholders.
Phase 1	D1 + API: register, me; dashboards show "Registered as X" and (users) link to quests.
Phase 2	Quest integration: quest list and progress; on-chain verification for faucet/first_tx.
Phase 3	Operator metrics: leaderboard and per-account blocks/uptime from indexer or RPC.

2.7 Community Quests

Community quests are user-facing tasks (e.g. "Use the faucet", "Send a transaction", "Share feedback") that qualify participants for testnet rewards. Completion can be **auto-verifiable on-chain** or **manual with proof**.

Quest types (examples)

Quest ID	Name	Verification	Reward tier

`faucet`	First drip	On-chain: faucet tx or balance > 0	Base user
`first_tx`	First transaction	On-chain: nonce ≥ 1 or tx in block	Base user
`validator_connect`	Join the network	Manual: node ID / multiaddr or screenshot	Validator
`feedback`	Share feedback	Manual: form + account ID	Bonus
`social`	Join community	Manual: Discord handle + account ID	Bonus
`docs`	Read and confirm	Manual: "I have read" + account ID	Base user

D1 tables (quests)

```
CREATE TABLE IF NOT EXISTS quests (
  id TEXT PRIMARY KEY,
  name TEXT NOT NULL,
  description TEXT,
  verification_type TEXT NOT NULL,
  reward_tier TEXT,
  active INTEGER DEFAULT 1,
  created_at TEXT
);

CREATE TABLE IF NOT EXISTS quest_completions (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  quest_id TEXT NOT NULL,
  account_id_hex TEXT NOT NULL,
  proof_type TEXT,
  proof_value TEXT,
  submitted_at TEXT NOT NULL,
  verified_at TEXT,
  rejected_reason TEXT,
  FOREIGN KEY (quest_id) REFERENCES quests(id)
);
CREATE INDEX IF NOT EXISTS idx_quest_completions_account ON
quest_completions(account_id_hex);
CREATE INDEX IF NOT EXISTS idx_quest_completions_quest ON quest_completions(quest_id);
CREATE UNIQUE INDEX IF NOT EXISTS idx_quest_completions_unique ON
quest_completions(quest_id, account_id_hex);
```

Quest API

Method	Path	Purpose
GET	`/api/quests`	List active quests
POST	`/api/quests/submit`	Submit a completion
GET	`/api/quests/status?account_id_hex=0x...`	Quest progress for account

Define incentive rules on a single **Incentivized Testnet Rules** page (e.g. `/incentivized-rules`); link from Quests and announcements.

Part 3 — Incentivized Testnet

Readiness, promotion, and mainnet migration for an incentivized testnet where validators, developers, and users can earn rewards. References: READINESS.md, RUNBOOK.md, BOING-BLOCKCHAIN-DESIGN-PLAN.md, BUILD-ROADMAP.md.

3.1 Readiness & Launch

Current Status

VibeMiner shows "no nodes" until bootnodes and public RPC are live. Complete the Launch-Blocking Checklist in READINESS.md before announcing the incentivized testnet.

Duration

Duration	Recommendation
1 week	Soft launch or rehearsal only
2–4 weeks	**Recommended** for first incentivized testnet
1 month+	Consider for Phase 2 after a 2–4 week Phase 1

Technical Readiness Checklist

- [ ] Bootnodes running; multiaddrs in this doc (§6) and website
- [ ] Public RPC live; faucet enabled and tested
- [ ] Genesis and chain ID documented; multi-node sync verified
- [ ] Website testnet/faucet pages show correct RPC and bootnodes

Incentive Program Design

Role	Primary incentive	Optional mainnet link
Validators	Testnet BOING (blocks) + leaderboard/uptime	Small allocation or NFT for top N (capped)
Developers	Testnet BOING, bug bounties, dApp recognition	Grant eligibility
Users	Faucet + quests/feedback	Optional NFT (capped)

Document caps and eligibility in a single "Incentivized Testnet Rules" page.

Pre-Launch Checklist

- [ ] Incentive program documented with criteria and caps
- [ ] **Testnet Portal** live: registration (developer / user / node operator), dashboards; see Part 2. Community quests live under **Users** at `/testnet/users`.
- [ ] Bug bounty scope (if any) defined; contact for submissions
- [ ] Feedback channel live (Discord, GitHub Discussions)

- [ ] Announcement draft with start/end dates and links

Launch Day

1. Publish bootnode list and public RPC URL
2. Announce start with duration and links
3. Monitor RPC and bootnodes
4. Pin quick start in Discord/Telegram

Phase 1 Parameters (fill before launch)

Parameter	Placeholder
Start date	e.g. 2025-03-15
End date	2–4 weeks recommended
Public RPC	https://testnet-rpc.boing.network/
Validator rewards	TBD — define criteria and caps
Developer rewards	TBD — per track
Bug bounty	GitHub Security Advisories or security@

3.2 Promotion

One-liner: "Earn testnet BOING and qualify for mainnet BOING. Run a validator, build a dApp, or complete quests — rewards from our Community & Grants pool."

Channels and timeline: Teaser 2–3 weeks before; publish full rules 1 week before; announce on launch day; link testnet, faucet, rules. Promote via Twitter/X, Discord, Telegram, blog, website, VibeMiner/partners. Link to the **Testnet Portal** (`/testnet`) for registration and dashboards.

3.3 Mainnet Migration

Before testnet ends: Fix end date; clarify "mainnet reward" meaning; collect validator addresses, bug reports, quest submissions.

After testnet ends: Thank-you post; publish results/leaderboards; no new promises; stick to published rules.

Mainnet launch: "Same chain you tested is now mainnet"; publish "Testnet rewards claim" page; distribute mainnet BOING from Community & Grants pool per rules. Reserve a defined **Testnet Rewards** slice (e.g. X% of initial supply) from **Community & Grants (15%)**; publish the cap.

3.4 Appendix A — Reddit Post Draft (Promotion)

Use this draft to promote the incentivized testnet on Reddit. Fill in **[PLACEHOLDERS]** (dates, links) before posting. Check each subreddit's rules and use the right flair.

Title options (pick one or adapt)

- **Boing Network — Incentivized testnet is live: earn testnet BOING as validator, developer, or user (L1 with protocol-enforced QA)**

- **We're running an incentivized testnet for Boing Network — an L1 where only quality deployments are accepted on-chain. Validators, devs, and users can earn rewards.**
- **Boing Network incentivized testnet: 2–4 weeks to run a validator, build a dApp, or complete quests — rewards from our Community & Grants pool**

Post body (short)

Boing Network is an L1 blockchain built from first principles with **protocol-enforced quality assurance**. We're the first chain where only deployments that meet defined security and compliance rules are accepted on-chain. We're launching our **incentivized testnet** and invite validators, developers, and users to participate. **Validators:** Earn testnet BOING for blocks and uptime; top performers can qualify for mainnet recognition. **Developers:** Testnet BOING, bug bounties, dApp recognition, grant eligibility. **Users:** Faucet, quests, feedback; optional rewards per rules. All rewards from our **Community & Grants pool**. Timeline: Start [DATE], End [DATE], 2–4 weeks. **How to start:** boing.network/testnet/join · boing.network/faucet · Build from source or use **VibeMiner** for one-click validator. Rules & rewards: [Link to Incentivized Testnet Rules when live]. Tech: Rust, HotStuff BFT, libp2p, custom VM, Sparse Merkle. CLI: `boing init`, `boing dev`, `boing deploy`; TypeScript SDK. Questions welcome below.

Links to include

- Website: https://boing.network
- Join testnet: https://boing.network/testnet/join
- Faucet: https://boing.network/faucet
- GitHub: https://github.com/chiku524/boing.network
- Testnet doc: https://github.com/chiku524/boing.network/blob/main/docs/TESTNET.md

Subreddit tips

- **r/cryptocurrency** / **r/CryptoCurrency:** Use flair; respect karma rules; no referral links.
- **Large general forums:** Keep it technical and factual; avoid hype.
- **r/altcoin:** Focus on participation, not price.
- Post once per subreddit; answer comments; add a short comment after posting to seed discussion.

Boing Network — Authentic. Decentralized. Optimal. Sustainable.