**Synergy Network Configuration Specification**

**Interconnectivity for Testnet Components**

**1. Overview**

This document defines the configuration requirements and interface connection details necessary to integrate the Synergy Testnet with its supporting components, including the Synergy Explorer, Synergy Utility Tool (CLI & GUI), and external APIs. These specifications enable real-time data exchange, transaction processing, and interaction with Synergy smart contracts.

**2. Core Configuration Values**

**2.1 RPC Endpoints**

* **Main JSON-RPC:** https://rpc.testnet.synergy-network.io
* **WebSocket (WSS):** wss://ws.testnet.synergy-network.io

**2.2 Chain Configuration**

* **Chain ID:** 338638
* **Network Type:** testnet
* **Bech32m Address Prefixes:** sYnQ, sYnU

**2.3 Contract Deployment Standard**

* Contracts are deployed using Hardhat with a target network labeled testnet
* Compilation Command: npx hardhat compile
* Deployment Command: npx hardhat run scripts/deploy.js --network testnet

**3. Synergy Explorer Configuration Requirements,** to connect and display real-time Synergy Testnet data.

**3.1 Backend Configuration**

* Configure backend service to connect to JSON-RPC endpoint.
* Poll for new blocks, transactions, and contract events every 5 seconds.
* Store indexed data in PostgreSQL.

**3.2 Frontend Configuration**

* Set API URL: http://localhost:PORT/api
* Enable modules:
  + Transaction Tracking
  + Address Search
  + Block Analysis
  + Real-Time Data Updates

**3.3 Required Environment Variables**

* REACT\_APP\_RPC\_ENDPOINT=https://rpc.testnet.synergy-network.io
* REACT\_APP\_CHAIN\_ID=338638
* REACT\_APP\_WS\_ENDPOINT=wss://ws.testnet.synergy-network.io

**4. Synergy Utility Tool (CLI + GUI) Configuration**

**4.1 Electron GUI Integration**

* Use IPC to connect to python\_bridge.py backend
* Environment settings for GUI preload:
  + - SYNERGY\_RPC\_ENDPOINT=https://rpc.testnet.synergy-network.io
    - SYNERGY\_CHAIN\_ID=338638
    - SYNERGY\_WS\_ENDPOINT=wss://ws.testnet.synergy-network.io

**4.2 CLI Tool Settings**

* Default config file path: ~/.synergy/config.json
* Expected config format:

{

"network": "testnet",

"rpc\_url": "https://rpc.testnet.synergy-network.io",

"chain\_id": 338638,

"ws\_url": "wss://ws.testnet.synergy-network.io"

}

**4.3 Required RPC Calls**

* eth\_blockNumber
* eth\_getBalance
* eth\_sendTransaction
* eth\_call
* eth\_getTransactionReceipt

**5. Developer Tools / API Requirements**

**5.1 Web3 SDK Integration**

* Install: npm install web3
* Connection snippet:

const Web3 = require('web3');

const web3 = new Web3('https://rpc.testnet.synergy-network.io');

**5.2 Contract Interface Integration**

const contract = new web3.eth.Contract(ABI, CONTRACT\_ADDRESS);

await contract.methods.<method>().call();

**5.3 Sample JSON-RPC cURL Request**

curl -X POST --data '{"jsonrpc":"2.0","method":"eth\_blockNumber","params":[],"id":1}' https://rpc.testnet.synergy-network.io

**6. Required for All Modules**

* RPC/WS endpoints must be referenced from a single config source (env or JSON).
* Contract addresses and ABI paths should be dynamically loaded where possible.
* Chain ID must be used for network validation to prevent mis-broadcasts.

**7. Next Steps**

* Link this configuration to genesis.json and token\_metadata.json for complete integration.
* Implement automated endpoint validation on startup.
* Publish this specification to the main Synergy Network documentation repository for developers and node operators.