ARIA Chain Node Installation Instructions

1. Prerequisites

Before starting, ensure your environment meets the minimum hardware and software requirements.

Hardware Requirements

CPU

4-8 Cores (Intel Core i7 or AMD Ryzen equivalent).

RAM

16GB Minimum (32GB recommended for high-load validators).

Storage

500GB+ NVMe SSD (Capacity will grow with chain state).

OS

Linux (Ubuntu 22.04 LTS or 24.04 recommended) or macOS.

Software Dependencies

Update your system and install the required development libraries:

Bash

```
sudo apt update && sudo apt install -y build-essential git clang curl libssl-dev llvm libudev-dev make protobuf-compiler
```

Install Rust and the WebAssembly (Wasm) target:

Bash

```
curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh source $HOME/.cargo/env rustup default stable rustup update nightly rustup target add wasm32-unknown-unknown --toolchain nightly
```

2. Build the ARIA Node

Clone the repository and compile the binary. Using the `--release` flag is mandatory for production environments to ensure optimal performance.

Bash

```
# Clone the repository
git clone https://github.com/aria-chain/aria-node.git
cd aria-node
# Build the binary
cargo build --release -features mainnet
```

3. Network Configuration (Chain Spec)

To join the ARIA network, you need the Chain Specification (`chainspec`) file. This ensures your node knows the genesis block and bootnode addresses.

01

Download the latest chainspec

`wget https://docs.ariachain.io/specs/aria-mainnet-raw.json`

02

Verify the file

Ensure the `protocolId` and `genesisHash` match the official network announcement.

4. Node Installation & Execution

Running a Full Node

A full node syncs the entire history of the blockchain and provides RPC access but does not participate in block production.

Bash

```
./target/release/aria-node \
--base-path /tmp/aria-data \
--chain ./aria-mainnet-raw.json \
--name "MyAriaFullNode" \
--rpc-cors all \
--rpc-methods unsafe \
--rpc-external \
--ethapi-debug
```

Since ARIA uses Frontier, enabling `--ethapi-debug` allows the node to serve Ethereum-compatible JSON-RPC requests (e.g., for MetaMask).

Running a Validator Node

If you intend to produce blocks, you must run as a validator and inject your session keys.

Start the node in validator mode:

Bash

```
./target/release/aria-node \
  --base-path /tmp/aria-validator \
  --chain ./aria-mainnet-raw.json \
  --validator \
  --name "AriaValidator_01"
```

Generate Session Keys: While the node is running, call the RPC to generate new keys:

Bash

```
curl -H "Content-Type: application/json" -d '{"id":1,
"jsonrpc":"2.0", "method": "author_rotateKeys", "params":[]}'
http://localhost:9944
```

1. *Record the result (hex string); you will need this to bond your tokens on the ARIA Portal.*

5. Post-Installation Verification

To ensure your Frontier-enabled node is functioning correctly, check the following:

- **Substrate Sync:** Check if `best block` is increasing in your logs.
- **EVM Compatibility:** Verify the Ethereum RPC endpoint is active.

Bash

```
curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method":"eth_blockNumber","params":[],"id":1}' http://localhost:9944
```

6. Identifying Your Node (Peer ID)

When the node starts, it generates a unique **Peer ID** based on its node key. This is essential for manual peering or if you are setting up a private sub-network within ARIA.

- **Locate your Peer ID:** Look for the following line in your startup logs: `Local node identity is: 12D3KooW...`
- **Manual Peering:** If your node is behind a firewall, other nodes can connect to you using: `--reserved-nodes /ip4/<Your-IP>/tcp/30333/p2p/<Your-Peer-ID>`

7. Telemetry & Monitoring

To help the ARIA community monitor the health of the network, we recommend connecting to the public telemetry server. This allows you to see your node's block height and latency in real-time.

Add the following flag to your execution command:

Bash

```
--telemetry-url 'wss://telemetry.ariachain.io/submit 0'
```