Guide to Becoming a Validator for the Kima Network

Testing Environment: Ubuntu 22.04

Minimum Requirements

- **Operating System:** Ubuntu 22.04
- **CPU:** 4vCPU (8vCPU recommended)
  - Must be an Intel XEON E-series or any other XEON supporting SGX-SPS (Server Platform Services). The motherboard must also support SGX.
  - Please See Appendix A
- **RAM:** 16GB (32GB recommended)
- **Storage:** 512GB HDD (1TB recommended)
- **Network**
  - Public static IP
  - **Open Ports:**
    - 22: SSH (Secure Shell) protocol
    - 26656: Cosmos app CometBFT gossiping port for consensus
    - 26657: Cosmos app CometBFT RPC port
    - 9090: Cosmos app gRPC port
    - 5051: TSS-ECDSA P2P port
    - 5052: TSS-EDDSA P2P port
    - 5053: TSS app EDDSA (Solana chain signer) gossiping port
    - 8081: TSS-ECDSA info address
    - 8082: TSS-ECDSA info address

Prerequisites Requirements

1. **curl**: This command-line tool is necessary for fetching the external IP and interacting with various web services. Ensure every team member has it installed and configured properly on their local systems.

2. **sudo** privileges: Each team member will need sudo privileges to install system packages and perform various secure operations on their machines. This is crucial for maintaining the security and integrity of your development environment.

3. **git**: Essential for version control and collaboration, git allows the team to clone necessary repositories and manage code changes effectively. Make sure all team members have git installed and are familiar with its basic commands.

4. **An SSH private key configured for GitHub**: Each member must have an SSH key set up and linked to their GitHub account for secure repository access. This is essential for cloning private repositories and contributing to your projects securely.

Additional Notes:

- If the team needs to access our private Git repository, please ensure each member has their SSH private key configured correctly. This involves generating an SSH key if they haven't already, adding it to GitHub account, and ensuring it's properly loaded into their SSH agent.
- If necessary, we can organize a session to assist anyone who hasn't set up their SSH keys or is unfamiliar with the process. It's crucial that everyone can access our repositories without any issues.

Installation Steps

1. Initial Setup:

- Download scripts from the Github repository to your server using any convenient method.
  - `git clone git@github.com:kima-finance/kima-external-validator.git`

2. Prepare Scripts:

- Make the files executable with
  - `sudo chmod +x setup-validator.sh`
  - `sudo chmod +x update-config.sh`
  - `sudo chmod +x run-validator.sh`

3. Prepare Environment:

1. Create a new empty `.env` file:
Use the `touch` command to create a new empty file named `.env`. If the file already exists, this command won't change its contents but will update its last modified time. Here's how to do it:

```
touch .env
```

2. Copy the contents of `.env.template` to `.env`:

The `cp` command copies the contents of one file to another. In this case, you're copying the contents of `.env.template` into `.env`. If `.env` already exists, this will overwrite it, so be sure you want to do this. Use the command like so:

```
cp .env.template .env
```

3. Edit the `.env` file with Nano:

The following command will open the `.env` file in Nano. If the file does not exist, Nano will start with an empty file:

```
nano .env
```

Inside Nano, use your keyboard to navigate and make changes.

4. Add the path to your SSH key:

Inside the `.env` file, you might need to add or update a line for your SSH key.

SSH_PRIVATE_KEY_PATH="/home/yourusername/.ssh/id_rsa"

Replace `/home/yourusername/.ssh/id_rsa` with the actual path to your SSH key. After adding or changing this line, save your changes and exit Nano (typically with `Ctrl + O` to save and `Ctrl + X` to exit).

5. Add blockchain networks configuration:

In the `.env` file, you will need to add configuration details for each blockchain network you intend to connect with. This include RPC AND WSS endpoints, for example:

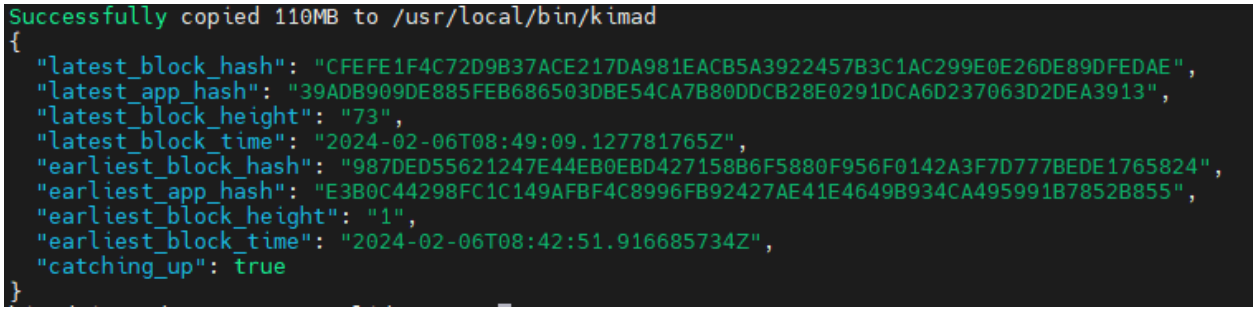
ETH_RPC_HOST="https://mainnet.infura.io/v3/YOUR_INFURA_API_KEY"
ETH_WSS_HOST="wss://sepolia.infura.io/ws/v3/ YOUR_INFURA_API_KEY "

Replace `"https://mainnet.infura.io/v3/YOUR_INFURA_API_KEY"` and `"https://bsc-dataseed1.binance.org"` with the actual network URLs and your API keys or relevant connection details. Again, save your changes and exit Nano.

For Testnet API keys, you can consider services like Alchemy, QuickNode, and Ankr as alternatives to Infura, providing support for various chains including Ethereum, Polygon, AVAX, BSC, Arbitrium, Optimism, Solana, and Tron. Additionally, each blockchain's official documentation may offer public testnet endpoints or other API services suitable for development and testing purposes.

Remember, these are templates, and you might need to adjust paths, keys, and configuration details based on your specific environment and requirements.

4. **Execute First Script:**
   - Run `./setup-validator.sh <validator-node-name>`
   - This script installs all necessary components and starts the software installation process.
   - After successful installation, the status will indicate as such:



5. **Synchronization:**

- Synchronizing with the blockchain network might take some time. Monitor the progress with `watch -n 1 'kimad status | jq .SyncInfo'`

- Once synchronization is complete, the `catching_up` status will indicate as such:

6. **Run Second Script:**
   - Run `./run-validator.sh`
   - It completes the validation node setup. Upon successful completion, you become a validator for the Kima Network blockchain.

Additional Information

Monitoring Containers: Use `docker ps` to see the status of the application container.

```
kimadmin@ci-node-2-development:~$ docker ps
CONTAINER ID   IMAGE                 COMMAND                  CREATED          STATUS                    PORTS
                    NAMES
b23b9f8277e6   kima_validator_kima   "/bin/sh -c ./entryp…"   27 seconds ago   Up 24 seconds (healthy)   0.0.0.0:1317→1317/tcp,
26656-26657→26656-26657/tcp   kima_validator_kima
```

Viewing Logs: To check the application logs, please replace the container ID field and execute `docker logs <CONTAINER ID> --tail 100`.

```
9:51AM INF committed state app_hash=2639067DF920CCB3EE5CED2F9E830BB1462F84EE81A2CC10B7FDA079DDF4F18B height=2310
9:51AM INF indexed block exents height=2310 module=txindex
9:51AM INF minted coins from module account amount=12150788uKIMA from=mint module=x/bank
9:51AM INF executed block height=2311 module=state num_invalid_txs=0 num_valid_txs=0
9:51AM INF commit synced commit=436F6D6D697449447B5B32313920333620323138203335203231302031363420383120382031333923
3239203834203937203533203132332033342032303950 3A3930377D module=server
9:51AM INF committed state app_hash=DB24DA23D2A451088B8B914DF3EA4DE64C9667BD4AAF2584BA1D5461357B22D1 height=2311
```

If you encounter any errors, please capture a screenshot and send it to us for assistance.

Appendix A

Kima SGX Hardware Requirements

Below is a list of hardware that’s compliant with Kima SGX requirements, updated to 2024.

Supported SGX Compliant CPU's List

CPU Model
Intel XEON E-2174G
Intel XEON E-2176G
Intel XEON E-2178G
Intel XEON E-2186G
Intel XEON E-2188G
Intel XEON E-2274G
Intel XEON E-2276G
Intel XEON E-2278G
Intel XEON E-2286G
Intel XEON E-2288G
Intel XEON E-2334G
Intel XEON E-2386G
Intel XEON E-2388G

Supermicro Servers list

Manufacturer	Motherboard Model
Supermicro	X11SCM-F
Supermicro	X11SCM-LN8F
Supermicro	X11SCW-F
Supermicro	X11SCZ-F
Supermicro	X11SSL-F
Supermicro	X11SCD-F
Supermicro	X11SCE-F
Supermicro	X11SCH-F
Supermicro	X11SCH-LN4F
Supermicro	X11SCL-F
Supermicro	X11SCL-LN4F
Supermicro	X12STW-TF
Supermicro	X12STW-F
Supermicro	X12STL-IF
Supermicro	X12STL-F
Supermicro	X12STH-SYS
Supermicro	X12STH-LN4F
Supermicro	X12STH-F

Manufacturer	Motherboard Model
Supermicro	X12STE-F
Supermicro	X12STD-F

Dell Servers list

Model
Dell R240
Dell R350

HP Servers list

Model
HP DL20 G10
HP DL20 G10+