

# Spin up a lite-node

Lite-nodes are a simplified node option that allow developers to perform lightweight tasks on a local node. This page covers how to spin-up a lite node on your local machine.

In this guide, we're going to use the [Lotus](#) Filecoin implementation. We'll show how to install a lite-node on MacOS and Ubuntu. For other Linux distributions, check out the [Lotus documentation](#). To run a lite-node on Windows, install [WLS with Ubuntu](#) on your system and follow the Ubuntu instructions below.

## Prerequisites

Lite-nodes have relatively lightweight hardware requirements – it's possible to run a lite-node on a Raspberry Pi 4. Your machine should meet the following hardware requirements:

1. At least 2 GiB of RAM
2. A dual-core CPU.
- 3.

To build the lite-node, you'll need some specific software. Run the following command to install the software prerequisites:

MacOS Ubuntu 1. Ensure you have [XCode](#) 2. and [Homebrew](#) 3. installed. 4. Install the following dependencies: 5.6. Copy 7. brewinstallgobzrqpkg-confighwloccoreutilsrust 8. 9. 1. Install the following dependencies: 2.3. Copy 4. sudoaptupdate-y 5. sudoaptinstallmesa-opengl-icdocl-icd-opengl-devgccgitbzrqpkg-configcurlclangbuild-essentialhwloc-libhwloc-devwget-y 6. 7. [Install Go](#) 8. and add/usr/local/go/bin 9. to yourPATH 10. variable: 11. 12. Copy 13. wgethttps://go.dev/dl/go1.21.1.linux-amd64.tar.gz 14. sudo rm -rf /usr/local/go && sudo tar -C /usr/local -xzf go1.21.1.linux-amd64.tar.gz 15. echo 'export PATH=PATH:/usr/local/go/bin' >> ~/.bashrc && source ~/.bashrc 16. 17. [Install Rust](#) 18. and source the ~/.cargo/env 19. config file: 20. 21. Copy 22. curl--proto=https--tlsv1.2-sSfhttps://sh.rustup.rs/sh 23. source "HOME/.cargo/env" 24. 25.

## Pre-build

Before we can build the Lotus binaries, there's some setup we need to do. MacOS users should select their CPU architecture from the tabs:

MacOS Intel MacOS ARM Ubuntu 1. Clone the repository, and move into the lotus 2. directory: 3.4. Copy 5. gitclonehttps://github.com/filecoin-project/lotus.git 6. cd lotus/ 7. 8. Switch to the branch representing the network you want to use. Mainnet always uses thereleases 9. branch: 10. 11. Copy 12. gitcheckoutreleases 13. 14. Or you can checkout to the Calibration testnet release using the twk/calibration 15. branch: 16. 17. Copy 18. gitcheckoutntwk/calibration 19. 20. Done! You can move on to the [Build](#) 21. section. 22. 1. Clone the repository, and move into the lotus 2. directory: 3.4. Copy 5. gitclonehttps://github.com/filecoin-project/lotus.git 6. cd lotus 7. 8. Switch to the branch representing the network you want to use. Mainnet always uses thereleases 9. branch: 10. 11. Copy 12. gitcheckoutreleases 13. 14. Or you can checkout to the Calibration testnet release using the twk/calibration 15. branch: 16. 17. Copy 18. gitcheckoutntwk/calibration 19. 20. Create the necessary environment variables to allow Lotus to run on M1 architecture: 21. 22. Copy 23. exportLIBRARY\_PATH=/opt/homebrew/lib 24. exportFFI\_BUILD\_FROM\_SOURCE=1 25. exportPATH="(brew--prefix)coreutils/libexec/gnubin:/usr/local/bin:PATH" 26. 27. Done! You can move on to the [Build](#) 28. section. 29. 1. Clone the repository, and move into the lotus 2. directory: 3. 4. Copy 5. gitclonehttps://github.com/filecoin-project/lotus.git 6. cd lotus 7. 8. Switch to the branch representing the network you want to use. Mainnet always uses thereleases 9. branch: 10. 11. Copy 12. gitcheckoutreleases 13. 14. Or you can checkout to the Calibration testnet release using the twk/calibration 15. branch: 16. 17. Copy 18. gitcheckoutntwk/calibration 19. 20. If your processor was released later than an AMD Zen or Intel Ice Lake CPU, enable the use of SHA extensions by adding these two environment variables. If in doubt, ignore this command and move on to [the next section](#) 21. . 22. 23. Copy 24. exportRUSTFLAGS="-C target-cpu=native -g" 25. exportFFI\_BUILD\_FROM\_SOURCE=1 26. 27. Done! You can move on to the Build section. 28.

## Build the binary

The last thing we need to do to get our node setup is to build the package. The command you need to run depends on which network you want to connect to:

Mainnet Calibration 1. Remove or delete any existing Lotus configuration files on your system: 2.3. Copy 4. mv~/lotus~/lotus-backup 5. 6. Make the Lotus binaries and install them: 7.8. Copy 9. makecleanall 10. sudomakeinstall 11. 12. Once the installation finishes, query the Lotus version to ensure everything is installed successfully and for the correct network: 13. 14. Copy 15. lotus--version 16. 17. This will output something like: 18.19. Copy 20. lotus version 1.19.1-dev+mainnet+git.94b621dd5 21. 22. 1. Remove or delete any existing Lotus configuration files on your system: 2. 3. Copy 4. mv~/lotus~/lotus-backup 5. 6. Make the Lotus binaries and install them: 7. 8. Copy 9. makeclean&&makecalibrationnet 10. sudomakeinstall 11. 12. Once the installation finishes, query the Lotus version to ensure everything is installed successfully and for the correct network: 13. 14. Copy 15. lotus--version 16. 17. This will output something like: 18. 19. Copy 20. lotus version 1.19.1-dev+calibrationnet+git.94b621dd5.dirty 21. 22.

## Start the node

Let's start the lite-node by connecting to a remote full-node. We can use the public full-nodes from [glif.io](#) :

Mainnet Calibration 1. Create an environment variable calledFULLNODE\_API\_INFO 2. and set it to the WebSockets

1. First, let's grab the head of the Filecoin network chain:
2. `'''`
3. Copy
4. `curl-XPOST'127.0.0.1:1234/rpc/v0\'`
5. `-H'Content-Type: application/json\'`
6. `--data'{"jsonrpc":"2.0","id":1,"method":"Filecoin.ChainHead","params":[]}'\`
7. `|jq`
8. `'''`
9. This will output something like:
10. `'''`
11. Copy

```

12. {
13. "jsonrpc": "2.0",
14. "result": {
15. "Cids": [
16. {
17. "/": "bafy2bzacead2v2y6yob7rkm4y4snthibuamzy5a5iuzlwvy7rynemtkdywfu0"
18. },
19. {
20. "/": "bafy2bzaced4zahevivrcdoefqlh2j45sevf5g3zsw6whpqxqjig6dxxf3ip6"
21. },
22. ...
23. ...
24. Next, let's try to create a new wallet. Since this is a privileged method, we need to supply our auth key eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbGxvdyl6WyJyZWFKliwid3JpdGUlLCJzaWduIiwiaWF0Ij0iXX0.eyJY7g-SDOsMheDRbQ9JlaFzus_Pan0J88VQ6ZLVE\
25. :
26. ...
27. Copy
28. curl-XPOST'127.0.0.1:1234/rpc/v0\
29. -H'Content-Type: application/json\
30. -H'Authorization: Bearer
    eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbGxvdyl6WyJyZWFKliwid3JpdGUlLCJzaWduIiwiaWF0Ij0iXX0.eyJY7g-SDOsMheDRbQ9JlaFzus_Pan0J88VQ6ZLVE\
31. --data '{"jsonrpc": "2.0", "id": 1, "method": "Filecoin.WalletNew", "params": [{"secp256k1"}]}'\
32. |jq
33. ...
34. This will output something like:
35. ...
36. Copy
37. {
38. "jsonrpc": "2.0",
39. "result": "t1vuc4eu2wgsdnce2ngygyzuxky3aqijqe7gj5qqa",
40. "id": 1
41. }
42. ...
43. The result field is the public key for our address. The private key is stored within our lite-node.
44. Set the new address as the default wallet for our lite-node:
45. ...
46. Copy
47. curl-XPOST'127.0.0.1:1234/rpc/v0\
48. -H'Content-Type: application/json\
49. -H'Authorization: Bearer
    eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbGxvdyl6WyJyZWFKliwid3JpdGUlLCJzaWduIiwiaWF0Ij0iXX0.eyJY7g-SDOsMheDRbQ9JlaFzus_Pan0J88VQ6ZLVE\
50. --data '{"jsonrpc": "2.0", "id": 1, "method": "Filecoin.WalletSetDefault", "params":
    [{"t1vuc4eu2wgsdnce2ngygyzuxky3aqijqe7gj5qqa"}]}'\
51. |jq
52. ...
53. This will output something like:
54. ...
55. Copy
56. {
57. "jsonrpc": "2.0",
58. "id": 1
59. }
60. ...
61.

```

## Next steps

You should now have a local lite-node connected to a remote full-node with an admin API key! You can use this setup to continue playing around with the JSON-RPC, or start building your applications on Filecoin!

[Previous Lite-nodes Next Fundamentals](#)

Last updated 4 months ago