Abelian Lite CLI Wallet Manual

Updated in December 2023

We will learn how to create an Abelian CLI (Command-Line Interface) lightweight wallet, use it to create wallet addresses, check balances and transfer ABELs.

This Abelian Lite CLI Wallet is basically the original Abelian CLI wallet (**abelwallet**), but instead of being configured to communicate with a local Abelian full node (**abec**) that you run, this Lite CLI Wallet is configured to communicate remotely with one of the Abelian full nodes that the Abelian Foundation or their partners are running.

Hence, you can make use of this Abelian Lite CLI Wallet to connect to any Abelian full node remotely as long as the remote node has some appropriate port (default port: 8667) opened and you have the needed RPC certificate and credential of the remote full node.

One obvious reason of using this Abelian Lite CLI Wallet instead of running the original pair locally, namely **abec** and **abewallet**, is the significant saving on disk storage and also the significant saving on time to wait for a local **abec** to do the synchronization. As of this writing, we can save more than 100GB of storage from running an **abec** normal node locally.

After getting the Abelian Lite CLI Wallet (https://www.abelian.info/downloads#5-abelian-wallet-cli) in compressed form, unzip it. All the commands below can be typed and executed in Terminal (macOS / Linux) or PowerShell (Windows).

1. Create a wallet

To create a wallet,

```
# macOS and Linux
./start_abewallet.sh --create
# Windows
abewallet --create
```

Note 1: On macOS, if it says `'xxx' cannot be opened because the developer cannot be verified`, go to `System Preferences -> Security & Privacy -> General` and click `allow anyway`. Then run abewallet as above again.

Note 2: On macOS (Apple silicon such as M1 and M2 chips / arm64), if it says `'xxx' is damaged and can't be opened`, type `xattr -d com.apple.quarantine path/to/xxx`, then run abewallet as above again.

Here is an example.

```
./start_abewallet.sh --create
Enter the private passphrase for your new wallet:
Confirm passphrase:
Enter the public passphrase for your new wallet:
```

```
Confirm passphrase:
NOTE: Use the --walletpass option to configure your public passphrase.
Do you have an existing wallet seed you want to use? (n/no/y/yes) [no]:
Your wallet generation seed is:
afe0f025646cde3eee099db9215f8cdb212ed0e06730fd0087e9d1ff5565fd53
the crypto version is 0
Your wallet mnemonic list is:
quiz,always,announce,silver,social,buyer,return,crisp,rice,april,tobacco,rent,base,half,light,toward,wonder,aerobic,whip,physical,vocal,real,vocal,leg
IMPORTANT: Keep the version and seed in a safe place as you
will NOT be able to restore your wallet without it.
Please keep in mind that anyone who has access
to the seed can also restore your wallet thereby
giving them access to all your funds, so it is
imperative that you keep it in a secure location.
Once you have stored the seed in a safe and secure location, enter "OK" to continue: OK
Creating the wallet...
Please remember the initial address:
00000000005a38589d86427698e3ec735335b368899ed6e0239c4285bbc4e370f4ea4e6d2ac1f1555b53f8df7c30e13d4bccc3b6d56763ec279620d9f131fb68089cb8ef18885950f84e56bf78d1780a5cac57d0888dabd669f86f85e7055afabae6a332fa000b3c6ee6a09751ce41ad7de4e...
```

The **public passphrase** will be used when launching abewallet,
and the **private passphrase** will be used to unlock the wallet (to transfer $ABEL).

Keep the **mnemonic list** in a safe place, it can be used to recover the wallet in the future in case you lose your wallet or want to set up your wallet at another computer.

The **initial address** (21,458 characters long) can be used as a mining address instance or payment address instance (for receiving ABELs) later. In this article, we will not discuss about the **short address** (132 characters long) and leave it to be covered in some application-layer tutorials.

2. Configure and run a live wallet

2.1 Configure a wallet

Before running a wallet, we need to change some of the settings in a configuration file called **abewallet.conf**. This configuration file is located at:

```
- Windows: C:\\Users\\username\\AppData\\Local\\Abewallet
- macOS: /Users/username/Library/Application Support/Abewallet
- Linux: /home/username/.abewallet
```

These folders are called **configuration folders**.

From https://www.abelian.info/downloads#6-abelian-lite-wallet-cli, you can download a pre-configured **abewallet.conf**. Below are the details of configuring **abewallet.conf**.

First, set **abecrpcuser** and **abecrpcpass** in **abewallet.conf**. For example, if you would like to connect to the Asia Site 1, you can set the following:

```
abecrpcuser=wgMURUUtK7o7pFgTg87D8GuSGg4=
abecrpcpass=biPPQXARkhtzQw3wV1NWAxuS6Vg=
```

If you would like to connect to other sites, you can set the following:

```
abecrpcuser=derKuJLKM5inu5uDmC7jxuHX57w=
abecrpcpass=ziUZ4ezY68ttB+9c7xBWmPQsIVg=
```

But remember that you can only connect to one site at a time. If you want to comment out any line in the configuration file, put a `;` symbol in the front of the line.

Next, set **rpcuser** and **rpcpass** in **abewallet.conf**:

```
rpcuser=XXX
rpcpass=YYY
```

where XXX and YYY are values chosen by you.

Then, set **rpcconnect** in **abewallet.conf** as follows.

```
rpcconnect=42.200.174.30
```

for connecting to Asia Site 1.

Please refer to the pre-configured **abewallet.conf** for other sites.

Finally, set the **cafile** in **abewallet.conf** by specifying the path of the full node RPC certificate **abec_rpc_tm.cert**.

```
cafile=~/path/to/abec_rpc_tm.cert
```

This full node RPC certificate can be downloaded from https://www.abelian.info/downloads#6-abelian-lite-wallet-cli. It is packed in a folder called **abec_certs/**. For example, we may put this folder in `~/Documents/` and set the **cafile** in **abewallet.conf** as

```
cafile=~/Documents/abec_certs/abec_rpc_tm.cert
```

At the above, `abecrpcuser` and `abecrpcpass` are used for establishing a secure RPC communication channel between your local **abewallet** with the remote full Abelian node with IP address specified in the parameter `rpcconnect`. While `rpcuser` and `rpcpass` are used for

establishing a local RPC communication channel between a wallet controller called **abewalletctl** (to be introduced) and this **abewallet**.

Currently, Abelian Foundation and their partners are running multiple full nodes (**abec**) and have them opened for everybody to connect to using their Abelian Lite CLI Wallet. Including the full node above, please find below the list of the full nodes with their IP addresses and the names of their RPC certificates. They all share the same `abecrpcuser` and `abecrpcpass` as above. You can choose one of them to connect to.

1. Asia Site 1: 42.200.174.30 abec_rpc_tm.cert
2. Asia Site 2: 112.120.106.200 abec_rpc_cblk.cert
3. Europe Site 1: 116.202.169.210 abec_rpc_cblk.cert
4. Europe Site 2: 3.76.69.82 abec_rpc_frankfurt2.cert
5. America Site 1: 3.132.2.26 abec_rpc_ohio2.cert
6. America Site 2: 52.86.138.73 abec_rpc_nvir2.cert

All these certificates are packed in the folder called **abec_certs/** and its compressed form can be downloaded from https://www.abelian.info/downloads#6-abelian-lite-wallet-cli.

2.2 Run a live wallet

After finishing the configuration above, run **abewallet** as below on a dedicated Terminal (macOS / Linux) or PowerSheel (Windows). The purpose of running **abewallet** constantly is to sync up the blocks from the remote Abelian full node for discovering all transactions related to your wallet address(es).

```
# macOS and Linux
./start_abewallet.sh --walletpass=[your public passphrase]
# Windows
abewallet --walletpass=[your public passphrase]
```

where `[your public passphrase]` is the public passphrase you had chosen when you created the wallet in Sec. 1.

Synchronization may take a while depending on the number of blocks on the Abelian network and the connection between your computer and the remote Abelian full node (say, several hours to one or two days) before completion, and we can see the status on the Terminal window. To find out the current block height of the Abelian network, please visit `https://explorer.abelian.info` .

3. Check balance

After **abewallet** has completed the synchronization as described above, run the following:

```
# macOS and Linux
./start_abewalletctl.sh --rpcuser=[rpcuser] --rpcpass=[rpcpass] --wallet getbalancesabe
# Windows
```

```
abewalletctl --rpcuser=[rpcuser] --rpcpass=[rpcpass] --wallet getbalancesabe
```

In the above, `[rpcuser]` and `[rpcpass]` are the values you chose when configuring the parameters **rpcuser** and **rpcpass**, respectively, in **abewallet.conf** in Sec. 2.1.

Example:

```
./start_abewalletctl.sh --rpcuser=XXX --rpcpass=YYY --wallet getbalancesabe
{"current_time":"2022-05-26 23:08:07.730291 +0800 SGT m=+10.601777429","current_height":12051,"current_block_hash":"00000000734e4c08b4422954da4df52c42fa1b3466e340ac332ad037f9a15ad5","total_balance":103424,"spendable_balance":102400,"freeze_balance":1024}
```

In the above, the field `total_balance` shows the balance of the wallet.

4 Create a new address instance for receiving ABELs

For ehancing your privacy, you may consider generating a new wallet address instance from the same wallet every time for receiving ABELs.

4.1 Unlock

Before creating a new address instance, we need to make sure that **abewallet** is running properly as described in Sec. 2.2. Next, unlock the wallet:

```
# macOS and Linux
./start_abewalletctl.sh --rpcuser=[rpcuser] --rpcpass=[rpcpass] --wallet walletunlock [private passphrase] [timeout]
# Windows
abewalletctl --rpcuser=[rpcuser] --rpcpass=[rpcpass] --wallet walletunlock [private passphrase] [timeout]
```

Example:

```
./start_abewalletctl.sh --rpcuser=XXX --rpcpass=YYY --wallet walletunlock myprivatepassphrase 240
```

This means unlocking the wallet with the private passphrase `myprivatepassphrase` for `240` seconds.

4.2 Create a new address instance

```
# macOS and Linux
./start_abewalletctl.sh --rpcuser=[rpcuser] --rpcpass=[rpcpass] --wallet generateaddressabe
# Windows
abewalletctl --rpcuser=[rpcuser] --rpcpass=[rpcpass] --wallet generateaddressabe
```

Example:

```
# macOS and Linux
./start_abewalletctl.sh --rpcuser=XXX --rpcpass=YYY --wallet generateaddressabe
[
  {
    "No": 2,
    "addr": "0000000000605660add742f3dcf88787a1618ecbc85289c188383d0df8ba292aa110b8592d3ef32da769bef62d707bad2dd870099fdd08cba30b52d6dcf4ca9c99834be04f2621b95ff8359ded46862fd229877a643951b66885fd3118c78b5d4773101a4e4a7e9a2c22780c15f29e1724edd9e9fa9c13c3d0483991b90fba4e9daeebeff42d0a4dcd93e4686cefceab6f5d07080ef1f77fdb9965354d48bbd82bfc7c47b8c295237b1ee88bd786b83931d7b74a8ba815ad14813d1f39b66a7c680bfdd77cce83aee4d018feac4928c0dc8f8774ee04ac60959be7f6a062f8c612c450b4957d5e1c48e3fa34a8bcdd6d93f1b70df0412dec32be232a5e54d4b0775ed031bc3dd585ad27e51b1e41662102ffc64e2b296224d8e4cf2cce9f26471b8b17bf26f9bc77985bc583098fe14c79473a3361866f513f43f859bc7492c4f6b6a7bcd2630a1669ba2e82e9ff93eecce1f4a77831f7742d12a966be4ad7f50b3313170465b649e5883861cda215d87ae801a1e11b5a5c...
```

The value (a long string with 21,458 characters) is a new wallet address instance being created. You can send this new address instance to a sender for him to send ABELs to you.

Note: The command above also tells you **the total number of addresses** that you have created so far. This number is also shown in the terminal window of **abewallet**.

Example:

```
2022-08-18 03:16:29.749 [INF] WLLT: The address with No. 2 is created.
2022-08-18 03:16:29.749 [INF] WLLT: Wallet status: current max No. of address is 2.
```

5. Transfer ABELs

Please refer to Sec. 4.1 for instructions on temporarily unlocking the wallet first.

As shown in Sec. 4.2, an ABEL wallet address instance is very long, and it's inconvenient to just directly copy and paste an address instance in the command line. As a result, what we are going to do is to create a file called **arg1** and store the file into a folder as below.

```
- Windows: C:\\Users\\username\\AppData\\Local\\Abec
- macOS: /Users/username/Library/Application Support/Abec
```

```
- Linux: /home/username/.abec
```

The content of *arg1* is as follows.

```
[
    {
        "address":"receiver1_address",
        "amount":10000000
    },
    {
        "address":"receiver2_address",
        "amount":20000000
    }
]
```

Note that the unit of amount is **Neutrino**. (1 ABEL = 10,000,000 Neutrinos). According to the content of the **arg1** above, we plan to send one ABEL to a receiver with address instance `receiver1_address` and 2 ABELs to `receiver2_address`.

After setting up the **arg1**, type the following to initiate a send transaction:

```
# macOS and Linux
./start_abewalletctl.sh --rpcuser=[rpcuser] --rpcpass=[rpcpass] --wallet sendtoaddressesabe -
# Windows
abewalletctl --rpcuser=[rpcuser] --rpcpass=[rpcpass] --wallet sendtoaddressesabe -
```

Example (on macOS):

```
$ cat /Users/username/Library/Application Support/Abec/arg1
[{"address":"receiver1_address", "amount":10000000},{"address":"receiver2_address", "amount":20000000 }]
$ ./start_abewalletctl.sh --rpcuser=XXX --rpcpass=YYY --wallet walletunlock myprivatepassphrase 240
$ ./start_abewalletctl.sh --rpcuser=XXX --rpcpass=YYY --wallet sendtoaddressesabe -
```

6. Recover a wallet

In case you need to run the wallet on another machine, you need to follow the instructions below to recover and set up your wallet on this new machine.

To recover a wallet, type

```
# macOS and Linux
./start_abewallet.sh --create
# Windows
```

```
4. abewallet --create
```

and after typing in **private passphrase**, **public passphrase**, type **yes** for the following question:

```
1. Do you have an existing wallet seed you want to use? (n/no/y/yes) [no]: yes
```

and type **0** for the following question:

```
1. Enter the crypto version is: 0
```

then enter your 24-word `mnemonic list` (a.k.a. recovery phrase) when the following prompt is out:

```
1. Enter existing wallet mnemonic:quiz,always,announce,silver,social,buyer,return,crisp,rice,april,t
   obacco,rent,base,half,light,toward,wonder,aerobic,whip,physical,vocal,real,vocal,leg
```

and finally enter **the total number of address instances to recover**. This is the total number of address instances for this particular address that have been created. Note that this number is the summation of the number you entered last time when recovering the wallet through the mnemonic list and the number of times that you have transferred ABELs out from this wallet since the last time that you have recovered the wallet. For example, if you entered ***1*** last time you recover your wallet through the mnemonic list and you made ***19*** ABEL transfers from this wallet. Then the total number of address instances to recover that you enter should be ***20*** here. Example:

```
1. Please input the max No. of address to recover : 20
```

Here is an example.

```
1. ./start_abewallet.sh --create
2. Enter the private passphrase for your new wallet:
3. Confirm passphrase:
4. Enter the public passphrase for your new wallet:
5. Confirm passphrase:
6. NOTE: Use the --walletpass option to configure your public passphrase.
7. Do you have an existing wallet seed you want to use? (n/no/y/yes) [no]: yes
8. Enter the crypto version is:0
9. Enter existing wallet mnemonic: quiz,always,announce,silver,social,buyer,return,crisp,rice,april,t
   obacco,rent,base,half,light,toward,wonder,aerobic,whip,physical,vocal,real,vocal,leg
10. Please input the max No. of address to recover :20
11. Creating the wallet...
12. 2022-06-09 10:06:15.069 [INF] WLLT: The addresses with No. in [0, 6] have been restored.
13. 2022-06-09 10:06:15.933 [INF] WLLT: Opened wallet
14. The wallet has been created successfully.
```

We understand that keeping track of this number can be troublesome. The Abelian team is now working hard on improving this privacy-preserving mechanism with the objective of eliminating this

address instance concept altogether so that we no longer need to keep track of the total number of address instances to recover in the near future.

For the timebeing, there is one command which can help us find out the current total number of wallet address instances:

```
# macOS and Linux
./start_abewalletctl.sh --rpcuser=[rpcuser] --rpcpass=[rpcpass] --wallet addressmaxsequence number
# Windows
abewalletctl --rpcuser=[rpcuser] --rpcpass=[rpcpass] --wallet addressmaxsequencenumber
```