Build a Filecoin mainnet mining cluster from scratch

lotus (https://learnblockchain.cn/tags/lotus)

Filecoin mining (https://learnblockchain.cn/tags/Filecoin%20%E6%8C%96%E7%9F%BF)

If you are a Filecoin novice, but have a little understanding of Linux operation and maintenance, then after reading this tutorial, you should be able to build a Filecoin mining cluster by yourself.

> The greatest luck in a person's life is to discover his mission halfway through his life, when he is still young and strong. - Zweig

Today's article is a supplement to the previous article "Filecoin Series 10-Building Lotus Testnet Cluster Mining" (http://www.r9it.com/20191217/lotus-cluster-mining.html) . The Filecoin mainnet has been launched, but the deployment details are slightly different from the previous testnet, so I wrote this article. If you are a Filecoin novice, but have a little understanding of Linux operation and maintenance, then after reading this tutorial, you should be able to build a Filecoin mining cluster by yourself.

Environmental description

- Operating system: Ubuntu-18.04LTS
- Golang version: 1.15.5
- Rust version: 1.46
- Lotus version: 1.2.1

Build a compilation environment

1. Install the go language environment and the dependent libraries required for compilation

```
sudo add-apt-repository ppa:longsleep/golang-backports
sudo apt update
sudo apt install golang-go gcc git bzr jq pkg-config mesa-opencl-icd ocl-icd-opencl-dev -y
```

The latest SDR acceleration code needs to be dependent `libhwloc` , so install it `libhwloc-dev` .

```
sudo apt install libhwloc-dev -y
# 有的系统还需要创建 lib 库软链接
ln -s /usr/lib/x86_64-linux-gnu/libhwloc.so.5.7.6 /usr/lib/x86_64-linux-gnu/libhwloc.so.15
```

Installation `golang-go` may be slow, if found to be particularly slow, it is recommended that you use the following method to manually install:

```
wget https://golang.org/doc/install?download=go1.15.5.linux-amd64.tar.gz # 这里选择你需要的版本
tar -xvpzf go1.15.linux-amd64.tar.gz
mv go /usr/local
```

Add environment variables: `vim ~/.bashrc`

```
export GOROOT=/usr/local/go
export GOBIN=$GOROOT/bin
export GOPKG=$GOROOT/pkg/tool/linux_amd64
export GOARCH=amd64
export GOOS=linux
export GOPATH=/golang/
export PATH=$PATH:$GOBIN:$GOPKG
```

Run the following command to make the environment variable take effect

```
source ~/.bashrc
```

2. Install Rust compilation environment

> Tip: This step requires a ladder. Without a ladder, the process may be very long.

```
bash curl https://sh.rustup.rs -sSf | sh
```

The above script will let you choose the installation method, directly select 1, and press Enter. After the installation is complete, run the following script to make the rust environment variable take effect:

```
source $HOME/.cargo/env
```

Compile lotus source code

1. Download the source code and switch to the designated branch.

```
# clone 源码到本地
git clone https://github.com/filecoin-project/lotus.git
# 切换到你想要分支或者 tag 版本，如：
cd lotus && git checkout v0.10.2
```

2. AMD machine compile command

```
FFI_BUILD_FROM_SOURCE=1 make clean all lotus-bench
```

3. For old Intel machines, it may be due to compatibility reasons (for example, an illegal instruction appears during execution: `illegal instruction`), you need to add the corresponding parameters:

```
FFI_BUILD_FROM_SOURCE=1 CGO_CFLAGS="-O -D__BLST_PORTABLE__" make clean all lotus-bench
```

Start lotus daemon

Because of `Lotus` the need to modify the default configuration of the nodes by setting environment variables of form, and use the global environment variable and easy to pollute the system environment, and sometimes lead to some baffling errors. So in order to avoid entering a lot of environment variables every time before using the lotus command , we recommend rewriting the lotus command.

First, create a new script

```
vim /usr/local/bin/lotus
```

The script content is as follows

```
export IPFS_GATEWAY=https://proof-parameters.s3.cn-south-1.jdcloud-oss.com/ipfs/
export LOTUS_PATH=/data/lotus
/opt/lotus/lotus $*
```

- `/data/lotus` lotus block storage address
- `/opt/lotus/lotus` lotus executable file path

Start lotus daemon

```
lotus daemon > daemon.log 2>&1 &
```

Note: The above command is to start full node synchronization blocks from zero, it will be slower, at current height `286186` , the network is good, I guess we have synchronized five days to one week.
But usually if you are not a wallet or exchange, you do not need a full node, you can directly synchronize from the official snapshot, or export the snapshot from other nodes, and then import it to the current daemon.
You can create a complete CAR snapshot in the following ways:

```
# usage
lotus chain export <filename>
# e.g
lotus chain export lotus-chain-2020-12-02.car
```

You can export a complete lotus chain snapshot through the above command. However, this snapshot is relatively large and may exceed 100 GB. So the more common way is to use the following command to export a castrated version of the snapshot, which is to export only the latest block and skip the old messages.

```
# usage
lotus chain export --skip-old-msgs --recent-stateroots=<stateroots-num> <filename>
# e.g
lotus chain export --skip-old-msgs --recent-stateroots=2000 lotus-chain-2020-12-02.car
```

`--recent-stateroots` The parameter specifies the number of state roots to be exported. The **minimum snapshot --recent-stateroots is 900. Snapshots smaller than 900 may fail to import** . `--skip-old-msgs` Parameter indicates export only those `stateroots` blocks directly referenced.

Then execute the import snapshot when the daemon is started for the first time. (It's just the first time you need to import it, you don't need to start the daemon later).

> Note: If the import fails, try to increase the `--recent-stateroots` value, but usually suffice 2000.

```
# Without verification
lotus daemon --import-snapshot <filename>
# With verification
lotus daemon --import-chain <filename>
```

If you want to quit immediately after the snapshot import daemon (for example, useful in the docker environment), add the command `--halt-after-import` flags:

```
lotus daemon --import-snapshot --halt-after-import <filename>
```

Initialize miner

Before performing this operation, you'd better wait for lotus daemon to complete block synchronization, use the following command to check whether the block synchronization is complete

```
lotus sync wait
```

This command will show how many blocks are currently synchronized, how many have not been synchronized, and your current synchronization speed. If the synchronization is not completed, it will always be in a blocked state, waiting for the synchronization to complete the command before exiting.

Similarly, we also need to rewrite `lotus-miner` the script

```
vim /usr/local/bin/lotus-miner
```

The script content is as follows

```
export FIL_PROOFS_PARAMETER_CACHE=/gamma/filecoin-proof-parameters  # proof 参数路径
export IPFS_GATEWAY=https://proof-parameters.s3.cn-south-1.jdcloud-oss.com/ipfs/
export LOTUS_PATH=/data/lotus  # daemon 数据根路径
export LOTUS_MINER_PATH=/data/lotus-miner   # miner 数据存储跟路径
export FIL_PROOFS_USE_GPU_COLUMN_BUILDER=1  # 启用 GPU加速 COLUMN hash 计算
export FIL_PROOFS_USE_GPU_TREE_BUILDER=1  # 启用 GPU 构造 Merkle 树
export RUST_LOG=Info
export FIL_PROOFS_MAXIMIZE_CACHING=1
/opt/lotus/lotus-miner $*
```

1. Create a wallet:

   ```
   lotus wallet new bls
   ```

   Output purse `f3xxxx` address: .

   Warning: Don't rush to recharge after creating a wallet, but back up the wallet private key first! ! !

   ```
   lotus wallet export f3xxxx
   ```

2. Recharge, if you just create a miner, recharge 1 FIL is enough.
3. Create miners.

```
# usage
lotus-miner init --owner=<address> --sector-size=<val>
# e.g
lotus-miner init --owner=f0xxxx --sector-size=32GiB
lotus-miner init --owner=f0xxxx --sector-size=64GiB
```

Modify Miner configuration

The configuration document of miner is in `$LOTUS_MINER_PATH/config.toml` . If you are running a single node, you do not need to modify the configuration document, but if you need to run a cluster,
then you need to modify the following places:

1. Modify API connection configuration, mainly used for worker connection miner.

```
# e.g
[API]
ListenAddress = "/ip4/xxx.xxx.xxx.xxx/tcp/2345/http"
RemoteListenAddress = "xxx.xxx.xxx.xxx:2345"
```

`xxx.xxx.xxx.xxx` Replace with your miner's intranet IP address.

2. Modify `[Storage]` option, all assigned tasks related to worker seal do.

```
[Storage]
#  ParallelFetchLimit = 10
AllowAddPiece = false
AllowPreCommit1 = false
AllowPreCommit2 = false
AllowCommit = false
#  AllowUnseal = true
```

Use remote lotus daemon

If your miner and lotus daemon are running on different machines, then you need to do some additional configuration:

1. Configure your lotus daemon config.toml file:

```
# e.g
[API]
ListenAddress = "/ip4/xxx.xxx.xxx.xxx/tcp/1234/http"
RemoteListenAddress = "xxx.xxx.xxx.xxx:1234"
```

`xxx.xxx.xxx.xxx` Replace with the intranet IP address of your lotus daemon.

2. Api lotus dameon copy of the file to the miner and the token machine `$LOTUS_PATH` directory.

Start lotus-miner

```
lotus-miner run > miner.log 2>&1 &
```

Start lotus-worker

Similarly, we also need to rewrite `lotus-worker` the script

```
vim /usr/local/bin/lotus-worker
```

The script content is as follows

```
export FIL_PROOFS_PARAMETER_CACHE=/gamma/filecoin-proof-parameters  # proof 参数路径
export IPFS_GATEWAY=https://proof-parameters.s3.cn-south-1.jdcloud-oss.com/ipfs/
export LOTUS_WORKER_PATH=/data/lotus-worker  # worker 数据存储目录
export LOTUS_MINER_PATH=/data/lotus-miner   # 保存 miner api 和 token 的目录
export FIL_PROOFS_USE_GPU_COLUMN_BUILDER=1  # 启用 GPU加速 COLUMN hash 计算
export FIL_PROOFS_USE_GPU_TREE_BUILDER=1  # 启用 GPU 构造 Merkle 树
export RUST_LOG=Info
export FIL_PROOFS_MAXIMIZE_CACHING=1
/opt/lotus/lotus-worker $*
```

1. Copy miner's api and token files to the `$LOTUS_MINER_PATH` directory.
2. Start the worker program, you can define the type of worker according to different parameters.

   (1) Start a worker that only accepts P1 tasks

   ```
   lotus-worker run --listen=11.11.11.11:2345 --precommit1=true --precommit2=false -commit=false
   ```

   Note: You `11.11.11.11` need to replace it with your worker's intranet IP address.

   (2) Start a worker that can accept P1 and P2 tasks at the same time

   ```
   lotus-worker run --listen=11.11.11.11:2345 --precommit1=true --precommit2=true -commit=false
   ```

   (2) Start a worker that only accepts C2 tasks

```
lotus-worker run --listen=11.11.11.11:2345 --precommit1=false --precommit2=false -commit=true
```

After the worker starts automatically through our `lotus-worker` API Configuration script, the miner is connected to receive the task, you can view a list of worker already connected to the miner by the following command.

```
lotus-miner sealing workers
```

Worker is also started, then we can start staking the sector and start mining.

```
lotus-miner sectors pledge # 质押一个随机数据的扇区，开始密封
```

Query the task distribution of the current cluster:

```
lotus-miner sealing jobs
```

General cluster operation and maintenance

For cluster operations, please refer to Filecoin Operation and Maintenance (2)-Environment Variables and Common Operation and Maintenance Operations (https://learnblockchain.cn/article/1465) , which will not be repeated here.

This article participates in the DingChain community writing incentive plan (https://learnblockchain.cn/site/coins) , good articles are good for profit, and you are welcome to join as well.

Published on 2021-04-20 21:02 Reading (306) Credits (5)
Category: FileCoin (https://learnblockchain.cn/categories/FileCoin)

1 like ♡ Reward Favorites

Articles you may be interested in

Talk about the optimization direction of Filecoin cluster from the three stages of cluster development (https://learnblockchain.cn/article/1998) 406 views

Time Cloud | Let's talk briefly about Filecoin Plus (https://learnblockchain.cn/article/1974) 306 views

Mining: Several key time points to invest in Filecoin (https://learnblockchain.cn/article/1885) 398 views

IPFS News: Understanding Filecoin's Circular Supply (https://learnblockchain.cn/article/1880) 356 views

Why the Filecoin project consumes Gas (https://learnblockchain.cn/article/1865) 302 views