Filecoin two-stage test (1)-small test

Build a Filecoin mining cluster
(https://learnblockchain.cn/tags/%E6%90%AD%E5%BB%BAFilecoin%20%E6%8C%96%E7%9F%BF%E9%9B%86%E7%BE%A4)

> The much-anticipated Phase 2 test of Filecoin has been in full swing for 2 weeks. Since the customer's machine is not in place, and the expensive price of the Filecoin miner is not affordable for our kind of dick, we only conducted the test this week.

Series navigation:
Filecoin two-stage test (1) (https://learnblockchain.cn/article/1191)
-small (https://learnblockchain.cn/article/1193)test
(https://learnblockchain.cn/article/1191)Filecoin two-stage test (2)
(https://learnblockchain.cn/article/1193)-AMD (https://learnblockchain.cn/article/1226)CPU
victory (https://learnblockchain.cn/article/1193)
Filecoin two-stage test (3)-heterogeneous cluster test
(https://learnblockchain.cn/article/1226)

He is strong by him, and the breeze is blowing on the hills;
he is horizontal by him, and the moon is shining on the river;
he is cruel, he is evil, and I am really angry.
---- Jin Yong, "The Legend of Slaying the Dragon"

I want to use the words of Mr. Jin to express my attitude towards the current Filecoin testnet's competition for hegemony. Be a pure tester, and test it quietly before the mainnet algorithm is confirmed or launched, so as to keep changing Just change.

The much-anticipated Phase 2 test of Filecoin has been in full swing for 2 weeks. Since the customer's machine is not in place, and the expensive price of the Filecoin miner is not affordable for our kind of dick, we only conducted the test this week.

Although the test results are not satisfactory, I still want to record the test results for the reference of students who are interested in joining the second-stage test.

In addition, we have before participating in the second stage, also the results of a test phase of the recorded video uploaded to `Bilibili` , the following is the video link:

In this test, we covered various test architectures. For example, in the mining architecture, we tested single-machine mining and cluster mining; in storage, we tested Ceph storage, Raid0, Raid5, and LVM; CPU we tested AMD and Intel,

GPUs use 2080Ti and 2070SUPER.

sector logic processing logic changes

The logic of the sector processing in the second phase of the test has some changes compared to the first phase. I think we have to understand these changes before officially starting the test.

First of all, I think the biggest change from the previous algorithm `window SDR` becomes `SDR` , as to what is `window SDR` and `SDR` that we explain later.

The second big change is the Sector Precommit process is divided into two stages, namely phase1 and phase2, that is, we have been talking about `阶段1(P1)` and `阶段2(P2)` .

The process of Phase1 is mainly a two-part calculation:

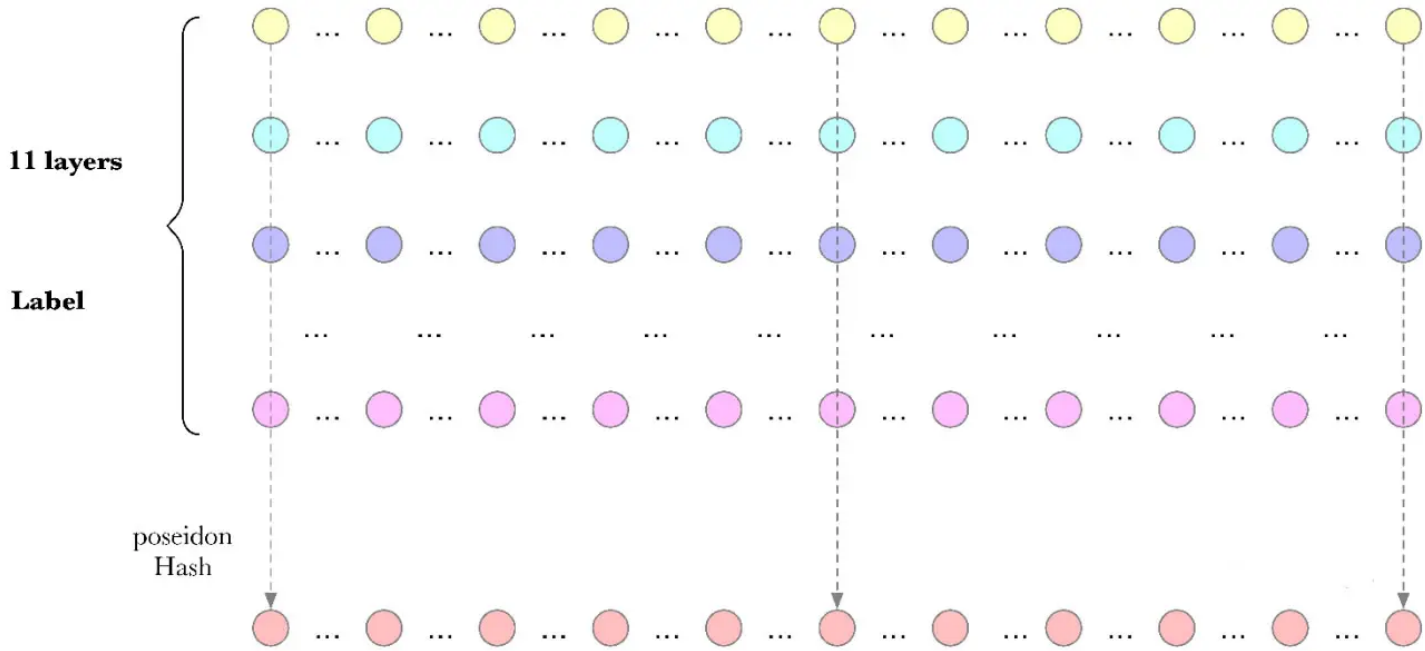
1. Calculate the merkle tree of the original data (binary tree, sha256 hash calculation)
2. label, which is the calculation of SDR. The merkle tree of the original data (tree_d), the root of the tree is comm_d.

The process of Phase 2 is mainly a two-part calculation

2. column hash

3. Generate a merkle tree (octree, poseidon hash calculation) for the calculation result of the column hash 3) Perform an encoding again on the calculation result of the label to generate a merkle tree (octree, poseidon hash calculation).

The calculation process of column hash is as follows:



1. The 32GB Sector is divided into 1G nodes. The calculation of SDR will generate 11 layers of processing data, each of which is 32GB. The node data of the same number on each layer,

Results of the combined hash is `column hash` the result of calculation. `Column hash` The calculated result is also 32GB.

2. For `column hash` calculation results, Octree (tree_c), root for the comm_c.
3. `label encoding` The calculation is to encode the SDR calculation result and the original data. The so-called encoding is currently the addition of large numbers.

As a result of encoding, an octree (`tree_r_last`) is generated , and the root of the tree is `comm_r_last` .

4. The data on the chain is two: `comm_d` and `comm_r` . Where `comm_r` it is `comm_c` and `comm_r_last` the hash result of posedion.

test environment

- System version: Ubuntu-18.04LTS
- Lotus version: `0.3.0'+gitf86a2ce'+api0.3.0`

Cluster test

In this cluster test, we also used the first stage of the 5 machines, that is, the memory was upgraded to 380GB. The specific configuration is as follows:

Miner configuration

- CPU: Intel(R) Xeon(R) CPU E5-2683 v4 @ 2.10GHz
- RAM: 512GB + 512GB Swap
- GPU: GeForce RTX 2080Ti x 1
- Storage: Ceph storage
- cache: 1TB SSD
- Network card: 10 Gigabit network card x 2

Worker configuration

- CPU: Intel(R) Xeon(R) CPU E5-2683 v4 @ 2.10GHz
- RAM: 380GB + 128GB Swap
- GPU: None
- Storage: 8TB enterprise hard drive
- cache: none
- Network card: 10 Gigabit network card x 1

Stand-alone test

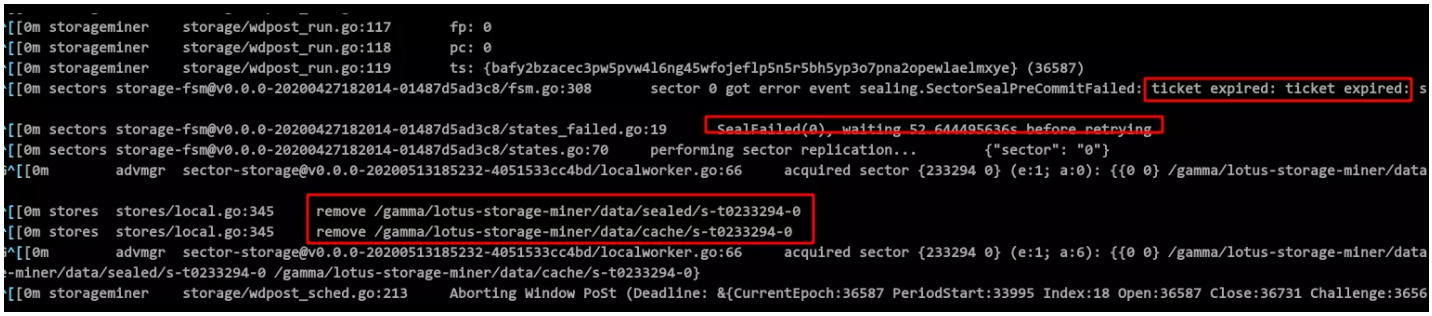
Another customer’s machine configuration is slightly lower, so we did a stand-alone test:

- CPU: Intel(R) Xeon(R) CPU E5-2680 v4 @ 2.40GHz
- RAM: 128GB + 128GB Swap
- GPU: GeForce RTX 2080Ti x 1
- Storage: Ceph storage
- cache: 1TB SSD
- Network card: 10 Gigabit network card x 1

Test Results

Almost none of the Intel machines we tested successfully submitted 1 sector. Although both slow done P1 and P2, but without exception have timed out, throws `ticket expire` an exception, which is not submitted completed within the specified time window.

```
2020-05-25T12:24:35.697Z        ^[[33mWARN^[[0m sectors storage-fsm@v0.0.0-20200427182014-01487d5ad3c8/f

2020-05-25T12:24:42.355Z        ^[[34mINFO^[[0m sectors storage-fsm@v0.0.0-20200427182014-01487d5ad3c8/s

2020-05-25T12:25:35.015Z        ^[[34mINFO^[[0m sectors storage-fsm@v0.0.0-20200427182014-01487d5ad3c8/s

2020-05-25T12:25:35.022Z        ^[[35mDEBUG^[[0m        advmgr  sector-storage@v0.0.0-20200513185232-405

2020-05-25T12:25:35.022Z        ^[[34mINFO^[[0m stores  stores/local.go:345     remove /gamma/lotus-stor

2020-05-25T12:25:35.023Z        ^[[34mINFO^[[0m stores  stores/local.go:345     remove /gamma/lotus-stor
```



In fact, it took almost 30 hours to run P1 on Intel's machines this time, and almost no Intel machines were successfully submitted individually, and at most were heterogeneous clusters of AMD and Intel.

Thread
#fil-lotus

askender May 23rd at 10:00 PM
Is there anyone still using Intel CPU? I found lotus mining (with Intel i9 9900) will get a error after precommit (spend 21 hours). After debuging, I make a pull request (https://github.com/filecoin-project/storage-fsm/pull/22) : previous thread: https://filecoinproject.slack.com/archives/CPFTWMY7N/p1590027349007500 Hope Intel CPU works as well as AMD one day in future. **Update: the pull-request makes Intel CPU work well. Proving now!** (edited)

> **askender**
>
> ```
> 2020-05-21T07:29:57.254+0800    WARN    sectors
> storage-fsm@v0.0.0-20200427182014-
> 01487d5ad3c8/fsm.go:308       sector 0 got error
> event sealing.SectorSealPreCommitFailed: ticket
> expired: ticket expired: seal height: 17814,
> head: 20927
> 2020-05-21T07:30:09.431+0800    INFO    sectors
> storage-fsm@v0.0.0-20200427182014-
> 01487d5ad3c8/states_failed.go:19
> SealFailed(0), waiting 47.569005836s before
> retrying
> ```
>
> How can I debug this problem? After this WARN, it rm stuff in `cache` and start precommit1 again.
>
> Thread in #fil-lotus | May 21st | View message

9 replies

John grema 3 days ago
My miner still working precommit1 over 24h.... intel xeon saclable silver...

John grema 3 days ago
It started precommit2 and then returned to precommit1.

askender 3 days ago
Yes. It will got a SectorSealPreCommitFailed error (you can search you log using the word: expire). My pull request can fix the `started precommit2 and then returned to precommit1` problem (edited)

John grema 2 days ago
Then how can use it?
Is it included in the latest version of Lotus?

askender 2 days ago
It just changes one line. you can vim that file yourself. It

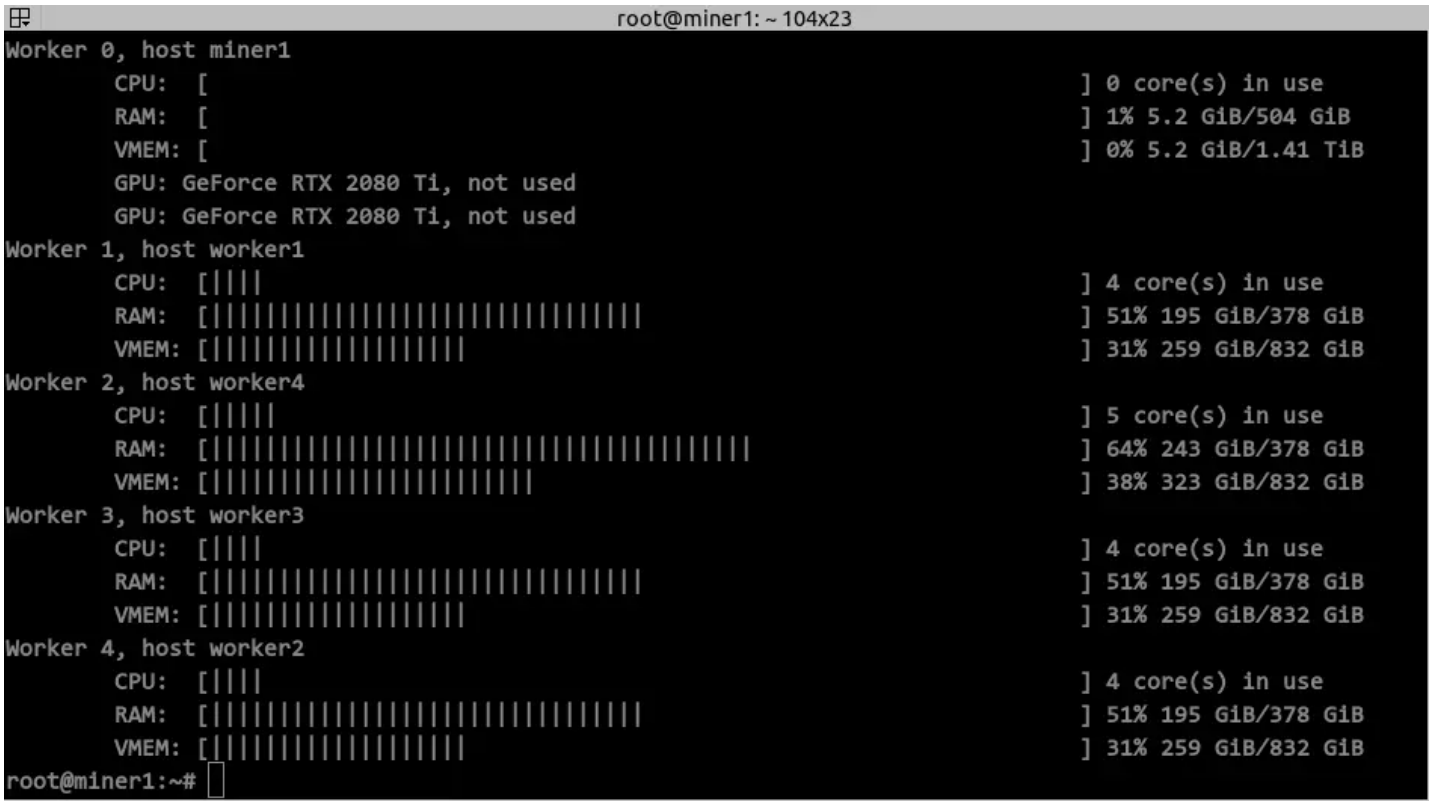
The test result data is as follows:

Machine configuration	Precommit 1	Precommit2
Cluster	36h5min	2h10min
Stand-alone	49h29min	3h5min

Small conclusion

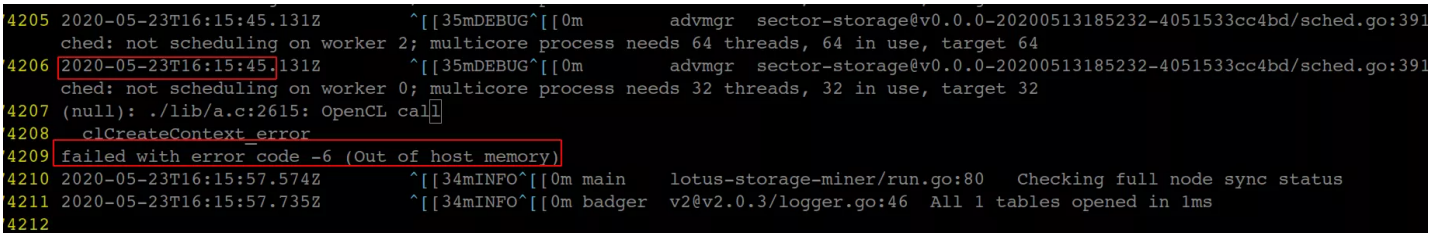
(1) Each sector seal can only occupy one CPU. In official words, **P1 can only skid on a single core** (the word "skid" is quite appropriate), so the slow speed of P1 is unbearable. Intel's machines basically take more than 30 hours, and the slightly worse machines are more than 40 hours.

(2) If you want to do Precommit in batches, you must have enough memory. At present, our 380GB memory can only seal 6 sectors at the same time. 256 GB is 3, and if there are more, it will report insufficient physical memory.



(3) Overall, Intel although do not very P1 row, barely rate `AMD 3960x` of one in five, but do P2 speed is good, with GPU acceleration, not much difference with AMD.

(4) If there are more Workers and you plan to use Miner for P2, the memory of Miner should be higher, otherwise the memory will overflow. Take our Miner as an example, 512GB RAM + 512GB Swap still hangs when doing 8 P2s at the same time.



Remember: Do not expect to add more Swap to what effect. According to our test observations, Swap uses up to 128GB, and generally does not use 64GB, and then I will report to you that there is insufficient memory or memory overflow. ********

The following is the formula for calculating the memory requirements of each stage of Lotus compiled by the miners for your reference:

No.	taskType	32G扇区所需资源:needRes				
		MaxMemory	MinMemory	Threads	CanGPU	BaseMinMemory
1	AddPiece	32G	32G	1		1G
2	PreCommit1	64G	64G	1		30G
3	PreCommit2	32G	32G	-1	TRUE	30G
4	Commit1	1G	1G	0		1G
5	Commit2	130G	30G	-1	TRUE	64G
6	Fetch	1M	1M	0	FALSE	0

最小内存需求：
minNeedMem := res.MemReserved + active.memUsedMin + needRes.MinMemory + needRes.BaseMinMemory
minNeedMem > res.MemPhysical

最大内存需求：
maxNeedMem := res.MemReserved + active.memUsedMax + needRes.MaxMemory + needRes.BaseMinMemory
maxNeedMem += MaxCachingOverhead // MaxCachingOverhead=32G
maxNeedMem > res.MemSwap+res.MemPhysical

线程需求：
多线程(Threads=-1)：cpu核心数需全部空闲才符合条件(SB条件)
其他(Threads=0、1)：cpu只要有1核心空闲都符合条件

GPU需求：只要存在GPU，且CanPU为true开启，并且GPU空闲

window SDR, SDR AND NSE

Here is a brief introduction `window SDR` , `SDR` and `NSE` , if you know about it , please skip it.

window SDR

A phase of testing enabled `Sector` copy proof algorithm. The window in the window SDR algorithm is the data of each Sector, which is first divided into 128M windows.

A 32G Sector will be divided into 256 windows. Window and window are independent of each other. For each window, SDR calculation (labeling encode) is performed separately. A total of 4 layers are required.

In other words, between independent windows, 4 layers of calculations are required. The relationship between layers meets the SDR.

SDR

Compared with window SDR, the algorithm of SDR is simpler and more rude. A Sector no longer divides the window. For the entire Sector, SDR is calculated, and 11 layers are calculated.

Post a log picture, let's feel it:

```
2020-06-01T03:08:10.124 INFO storage_proofs_porep::stacked::vanilla::proof >   generated layer 4 store with id layer-4
2020-06-01T03:08:10.125 INFO storage_proofs_porep::stacked::vanilla::proof > generating layer: 5
2020-06-01T03:32:21.661 INFO storage_proofs_porep::stacked::vanilla::proof >   setting exp parents
2020-06-01T03:32:25.220 INFO storage_proofs_porep::stacked::vanilla::proof >   storing labels on disk
2020-06-01T03:33:50.691 INFO storage_proofs_porep::stacked::vanilla::proof >   generated layer 5 store with id layer-5
2020-06-01T03:33:50.692 INFO storage_proofs_porep::stacked::vanilla::proof > generating layer: 6
2020-06-01T03:56:43.197 INFO storage_proofs_porep::stacked::vanilla::proof >   setting exp parents
2020-06-01T03:57:10.209 INFO storage_proofs_porep::stacked::vanilla::proof >   storing labels on disk
2020-06-01T03:58:36.787 INFO storage_proofs_porep::stacked::vanilla::proof >   generated layer 6 store with id layer-6
2020-06-01T03:58:36.788 INFO storage_proofs_porep::stacked::vanilla::proof > generating layer: 7
2020-06-01T04:22:58.063 INFO storage_proofs_porep::stacked::vanilla::proof >   setting exp parents
2020-06-01T04:23:05.656 INFO storage_proofs_porep::stacked::vanilla::proof >   storing labels on disk
2020-06-01T04:24:29.030 INFO storage_proofs_porep::stacked::vanilla::proof >   generated layer 7 store with id layer-7
2020-06-01T04:24:29.031 INFO storage_proofs_porep::stacked::vanilla::proof > generating layer: 8
2020-06-01T04:49:43.720 INFO storage_proofs_porep::stacked::vanilla::proof >   setting exp parents
2020-06-01T04:49:46.345 INFO storage_proofs_porep::stacked::vanilla::proof >   storing labels on disk
2020-06-01T04:51:09.545 INFO storage_proofs_porep::stacked::vanilla::proof >   generated layer 8 store with id layer-8
2020-06-01T04:51:09.546 INFO storage_proofs_porep::stacked::vanilla::proof > generating layer: 9
2020-06-01T05:15:27.132 INFO storage_proofs_porep::stacked::vanilla::proof >   setting exp parents
2020-06-01T05:15:29.742 INFO storage_proofs_porep::stacked::vanilla::proof >   storing labels on disk
2020-06-01T05:16:52.068 INFO storage_proofs_porep::stacked::vanilla::proof >   generated layer 9 store with id layer-9
2020-06-01T05:16:52.069 INFO storage_proofs_porep::stacked::vanilla::proof > generating layer: 10
2020-06-01T05:41:27.541 INFO storage_proofs_porep::stacked::vanilla::proof >   setting exp parents
2020-06-01T05:41:34.234 INFO storage_proofs_porep::stacked::vanilla::proof >   storing labels on disk
2020-06-01T05:43:00.982 INFO storage_proofs_porep::stacked::vanilla::proof >   generated layer 10 store with id layer-10
2020-06-01T05:43:00.983 INFO storage_proofs_porep::stacked::vanilla::proof > generating layer: 11
2020-06-01T06:05:49.669 INFO storage_proofs_porep::stacked::vanilla::proof >   setting exp parents
2020-06-01T06:06:13.547 INFO storage_proofs_porep::stacked::vanilla::proof >   storing labels on disk
2020-06-01T06:07:40.856 INFO storage_proofs_porep::stacked::vanilla::proof >   generated layer 11 store with id layer-11
2020-06-01T06:07:43.241 INFO filcrypto::proofs::api > seal_pre_commit_phase1: finish
```

NSE

NSE is the latest PoREP algorithm. The full name of the NSE algorithm: Narrow Stacked Expander PoRep.

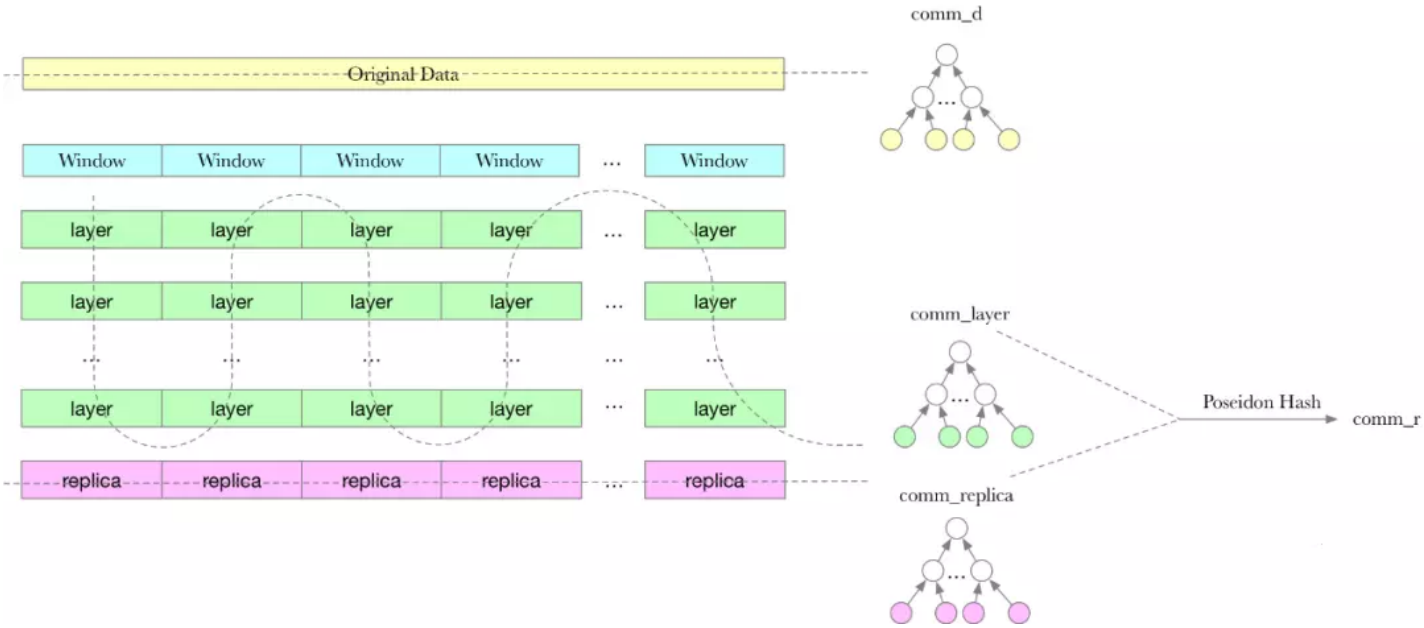
Source address: https://github.com/filecoin-project/rust-fil-nse-gpu (https://github.com/filecoin-project/rust-fil-nse-gpu)

NSE is called NSE because N, Narrow. Narrow means narrower than the previous SDR algorithm, and the data processed each time is a Window.

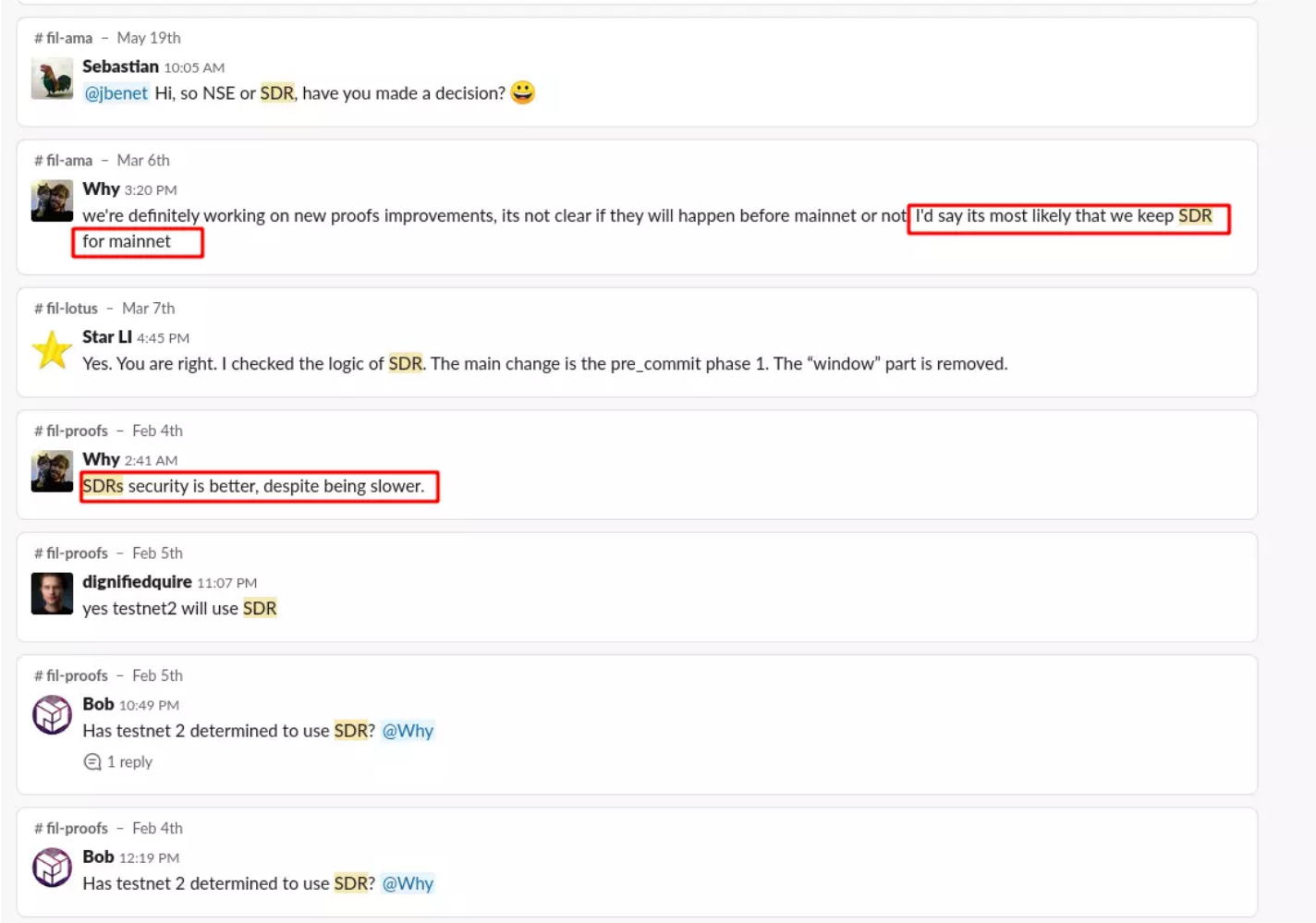
After each Window is processed layer by layer, a corresponding Replica will be generated. The data of each layer corresponding to all windows are constructed together into a Merkle tree. The data of all the Replica corresponding to the Window is also constructed into a Merkle tree.

The roots of two trees `Poseidon Hash` as a result of comm_r. comm_d and comm_r are the data that needs to be chained.

The entire algorithm flow is shown in the figure below:



What needs to be mentioned here is that it seems that NSE will not be activated before the mainnet goes live.



Test and tuning suitable for Xiaobai

There are many miners on Slack who shared how to optimize mining efficiency, but most of them involve code changes. Although some of them are rarely changed, those people claim that it is easy to change. But if you change it, you will find that it is not that easy.

In fact, the code of Lotus is extremely complicated. If you don't understand the entire mining process, you have to look at the code for several weeks or even months before you dare to change the core process (of course, you don't need to change the comment user or parameters. said).

Share tips here two in the testing process, absolutely **simple, the effect is visible** .

(1) Turn off Numa. This method is generally only useful for Intel machines. The Numa architecture is used to solve the performance problems caused by the shared BUS under the multi-physical CPU (non-multi-core) system. But as we said earlier, P1 only allows single-core processing, so there is no need to enable Numa at this time.

To turn off Numa, you need to turn it off in the Bios settings.

(2) Turn on the performance mode of the CPU. The default CPU in the Ubuntu system is working in `powersave` (power saving) mode, so the performance is not maximized, so it needs to be adjusted to `performance` (high performance) mode. You can adjust it temporarily by setting Bios or by software.

1. Install cpufrequtils:

```
1   sudo apt-get install cpufrequtils
```

2. View the current state of the cpu:

```
1   cpufreq-info
```

3. Adjust the cpu to performance mode:

```
1   sudo cpufreq-set -g performance
```

The above methods are available for pro-test, and the performance can be improved by about 5%.

> This article was first published at: Original link: http://www.r9it.com/20200525/filecoin-phase2-test1.html (http://www.r9it.com/20200525/filecoin-phase2-test1.html) 小一辈无产阶级码农 (http://www.r9it.com/) (http://www.r9it.com/20200525/filecoin-phase2-test1.html)

Reference link

- https://mp.weixin.qq.com/s/VkwDVcPJ753dJtHh8WtIow (https://mp.weixin.qq.com/s/VkwDVcPJ753dJtHh8WtIow)
- https://mp.weixin.qq.com/s/unqfaIdcWD8HAHtH1nSc2Q (https://mp.weixin.qq.com/s/unqfaIdcWD8HAHtH1nSc2Q)
- ubuntu adjusts the cpu operating mode to high performance (https://blog.csdn.net/li528405176/article/details/82823922)

Published on 2020-06-30 15:27 Reading (1528) Credits (120)
Category: FileCoin (https://learnblockchain.cn/categories/FileCoin)