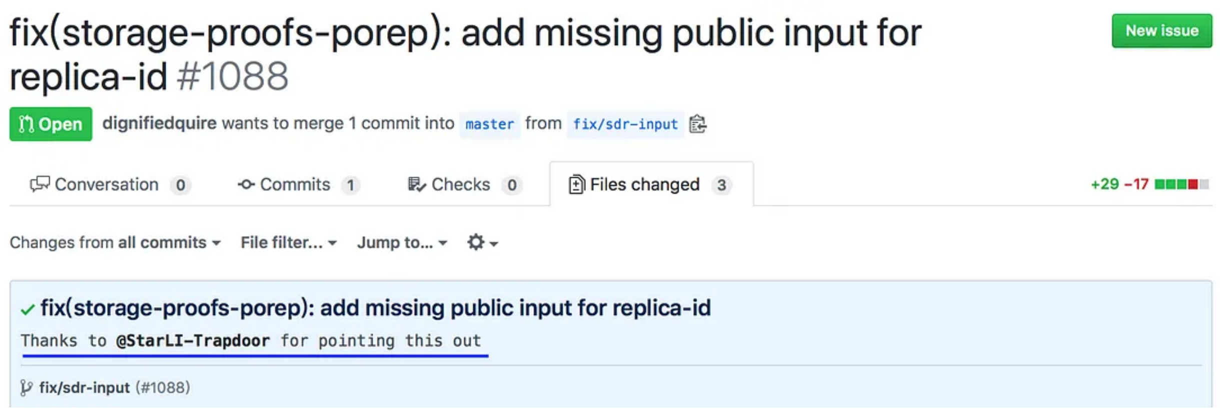
Filecoin —one PoREP vulnerability found by Trapdoor Tech

Trapdoor-Tech · Follow

4 min read · May 7, 2020

Listen Share

Trapdoor Tech discovered a serious vulnerability of the PoREP circuit (V25). Using this vulnerability, the calculation of SDR (Precommit1) can be directly omitted. Only one copy of all Sector Replica data is required. After the Trapdoor team communicated with the official in the first time, the official has quickly submitted the patch:



fix(storage-proofs-porep): add missing public input for replica-id #1088

Open dignifiedquire wants to merge 1 commit into `master` from `fix/sdr-input`

✓ fix(storage-proofs-porep): add missing public input for replica-id

```
Thanks to @StarLI-Trapdoor for pointing this out
```

fix/sdr-input (#1088)

The PoREP circuit has also been upgraded from V25 to V26. This article carefully talks about the attack principle of this serious vulnerability.

As we all know, the Sector data will be calculated through Labeling, Column Hash and Encoding to generate the final Replica data (Precommit1 and Precommit2 stage). The whole calculation generates a proof through zero-knowledge proof and submits it to the chain. It is not realistic to prove the processing of all the node data in the Sector, and the corresponding circuit will be very large. The current implementation of Lotus is to randomly check the data of 144 nodes and make the processing of the data of these 144 nodes into a circuit. Even so, the circuit scale is already very large, exceeding 100 million.

replica_id introduction

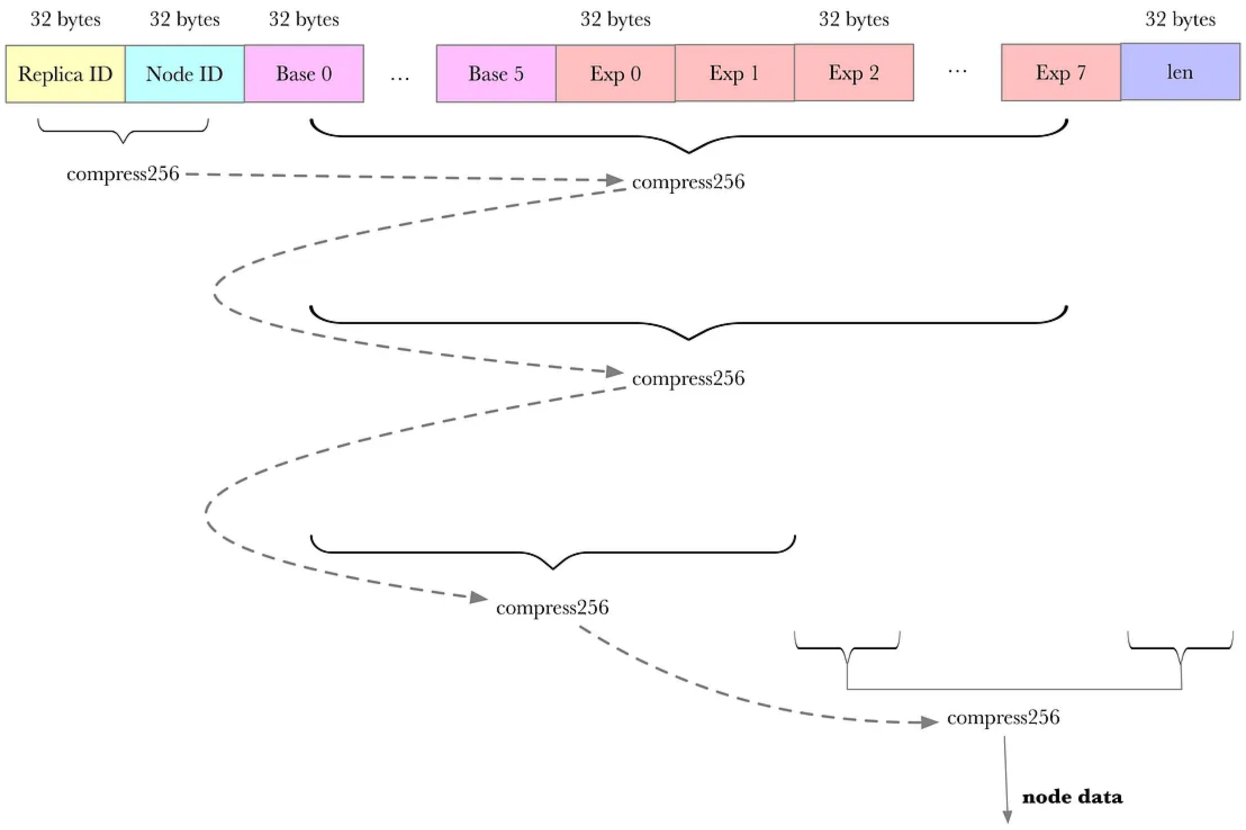
The replica_id is the label of each Replica data. The calculation method is as follows:

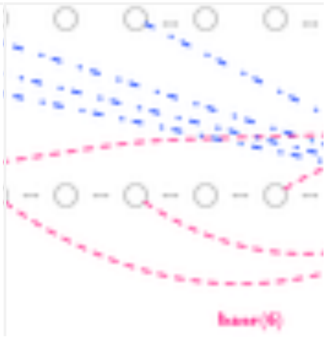
```
let replica_id =
    generate_replica_id::<Tree::Hasher, _>(&prover_id,
sector_id.into(), &ticket, comm_d);
```

In other words, replica_id is the hash result of provider_id, sector_id, comm_d and the random number ticket on the chain. Among them, prover_id is the id of the miner, sector_id is the number of the sector, which increases in sequence, and comm_d is the root of the merkle tree of the original data of the sector. It is easy to find that the Replica_id corresponding to each Sector is different.

SDR algorithm calculation

The calculation process of the SDR algorithm is also the process of Labeling. The calculation of each node of SDR requires a replica_id. The specific calculation process is as follows:





Filecoin — Deep into SDR algorithm

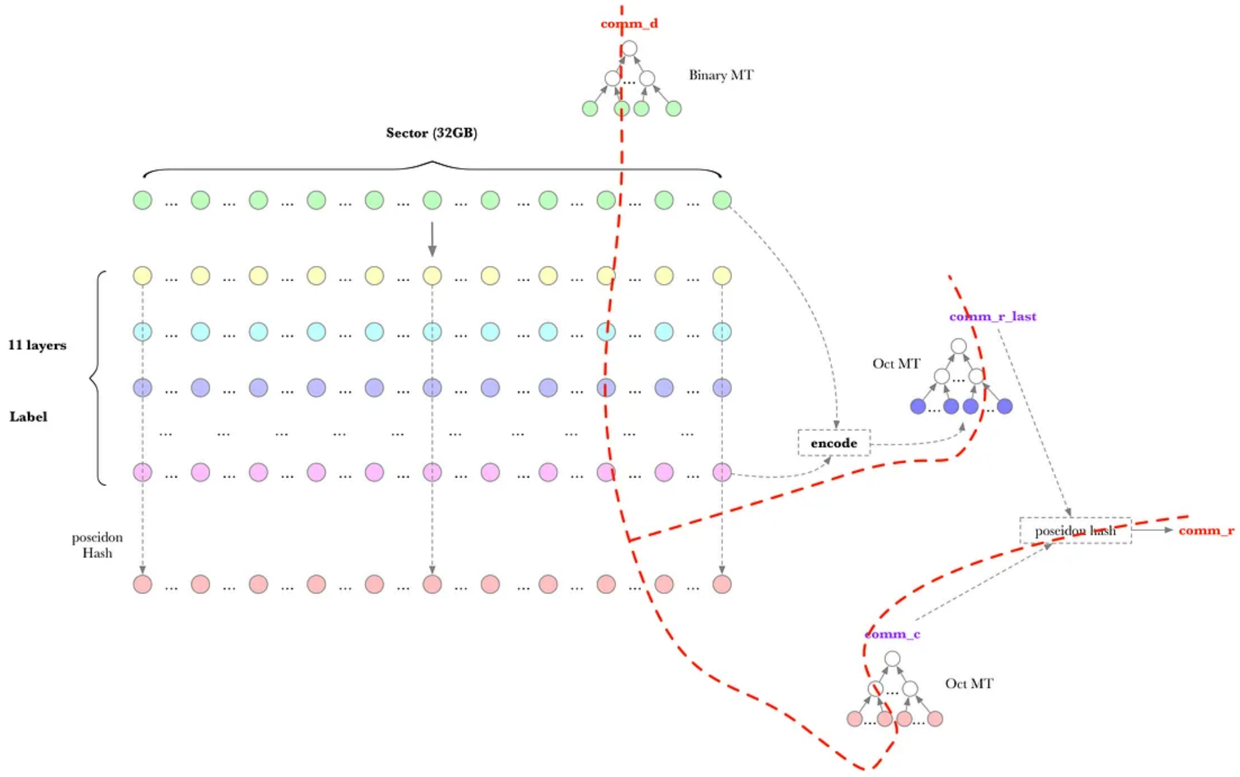
The algorithm processed by Lotus Sector has switched from window SDR to SDR. The so-called SDR is Stacked DRG...

medium.com

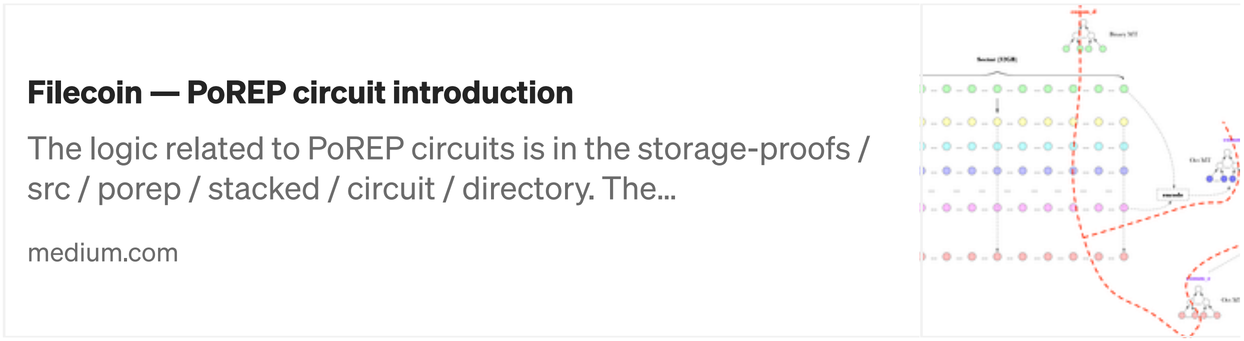
PoREP Circuit

After the Sector performs Labeling, Column Hash and Encoding calculations, it randomly selects 144 node data generation circuits and performs zero knowledge proof. The circuit logic corresponding to each node is as follows:

Each node needs to be correctly calculated by Labeling, Column Hash and Encoding. And the root of the corresponding Merkle trees should be generated from leaf.



The detailed explanation of PoREP circuit can be found at:



Filecoin — PoREP circuit introduction

The logic related to PoREP circuits is in the storage-proofs / src / porep / stacked / circuit / directory. The...

medium.com

Challenge nodes selection

The challenge nodes selection logic is implemented in the derive_internal function of the LayerChallenges structure in storage-proofs / porep / src / stacked / vanilla / challenges.rs:

```
let j: u32 = ((challenges_count * k as usize) + i) as u32;
let hash = Sha256::new()
    .chain(replica_id.into_bytes())
    .chain(seed)
    .chain(&j.to_le_bytes())
    .result();
```

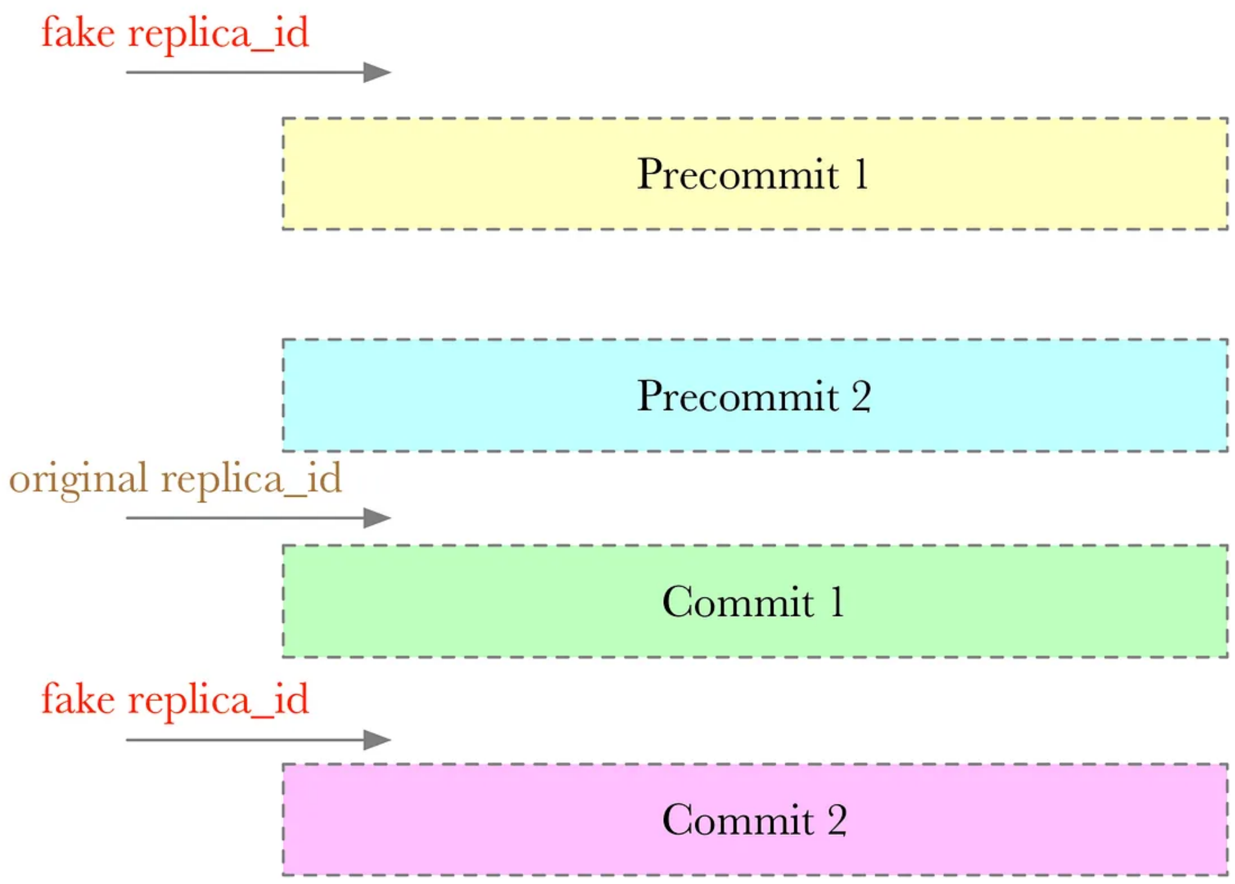
That is to say, the challenge nodes are generated by the hash calculation result of replica_id, the random number and challenge index. Simply put, the challenge nodes selection logic is related to replica_id.

Public input of PoREP circuit

In addition to comm_d and comm_r, the public input of the proof circuit of the entire PoREP also has path information of each Merkle tree. That is, the entire circuit defines the location of the spot check node, the comm_d of the original data and the comm_r of the final Replica data. It only limits this information.

What is the vulnerability?

Careful analysis of the public input of the PoREP circuit found that the public input of the PoREP circuit did not include replica_id. Although there is an indirect relationship between the location of the spot check node and the replica_id, in the case where the replica_id is not used as a public input, the SDR calculation can use a fake replica_id. Simply put, in the calculation process, any replica_id can be used for Labeling calculation. From this, the attack method is clear:



Because the calculation of Labeling is only related to replica_id, in the case of forged replica_id, the calculation of the entire SDR only needs to be calculated in advance. In other words, under the attack of this vulnerability, the SDR calculation is not needed.

All in all, the current circuit just proves SDR calculation is bounded with one replica_id, but not a specified replica_id. Simply put, one fixed replica_id can be used for SDR calculation. That's to say, all sectors can share one SDR replica.

To sum up:

PoREP V25's circuit does NOT use "replica_id" as public input. And the circuit only proves that SDR calculation is bounded with one replica_id, but not a specified replica_id. To exploit the vulnerability, one fixed replica_id can be used for all sectors' SDR calculation. That's to say, only one SDR replica is needed for all sectors.