ndau Incident Report – 2019-06-12

What happened?

On 10 June 2019 at 4:30 PM EDT it was reported that the ndau blockchain rejected the last of 5 transactions submitted in succession by ndev technical staff to credit accrued EAI to ndau holder accounts (`CreditEAI` transactions). After a few minutes of investigation it became clear that the blockchain was no longer processing blocks or transactions.

Why wasn't it processing blocks?

The ndau blockchain network currently has five validator nodes. The blockchain stopped reaching consensus at block 9574. 3 nodes were voting on a block with one hash, and 2 nodes were voting on a block with a different hash. No block had 2/3 of the voting power, and the blockchain could not proceed. This behavior is a fundamental feature of ndau's Tendermint consensus system, which prioritizes safety over consistency.

Why didn't the nodes agree?

The processing code performing the calculations to credit EAI had a subtle bug that created non-deterministic behavior. Conditions that triggered the bug were:

- Multiple source accounts were redirecting their EAI to the same destination account
- Those source accounts were all delegated to the same ndau node
- The amounts of accrued EAI and the elapsed time periods involved were small

Due to a feature of the Go programming language the source accounts were not always processed in the same order. Different nodes would use a different order. Each time EAI is credited to a destination account the Weighted Average Age (WAA) of that account is updated. The order of computation should not matter, but since each step rounds the result to the least significant digit the cumulative result could differ if the order was different. Since the amounts and time periods were very small, evaluating the source accounts in a different order resulted in a discrepancy of 1 microsecond (1 bit) in the computed WAA. This difference changed the hash of the block being validated: some nodes calculated one hash and some calculated a different one.

What was the mitigation process?

Identify the problem

At about 6:30 PM EDT our engineers had identified that we had a consensus failure problem due to a hash mismatch and had found the code responsible.

Fix the code

By 7:30 PM we mostly understood the cause and had a tentative fix that would prevent it from recurring. The fix was for all nodes to process accounts in a well-defined order for this transaction.

This fix was reviewed by other engineers and merged to our deployment branch.

Find a way to restart the blockchain

Because the transaction had been propagated but not committed, we didn't have an easy way to prevent the blockchain from continuing to try to process it.

We had a backup of the chain from block 9479. When comparing that backup to the state of the blockchain when it halted we saw that no transactions had occurred between block 9479 and the start of this group of `CreditEAI` transactions at block 9567.

Our primary goal was to get the blockchain up and running again as quickly as possible, so we decided that our best choice was to restart the blockchain from block 9479. This was easy for us to do ourselves, and we knew it would not cause any transactions to be rolled back (other than the ones submitted by ndev ourselves that caused the problem).

. *Note: we regret taking this approach; our primary goal should have been to retain complete transparency about the status of the ndau blockchain and the cause of the failure rather than to restore operation as quickly as possible.*

Test the fix

We built the new version of the code and deployed it to our testnet, which has a nearly identical configuration to our mainnet. We then restarted all 5 of the validator nodes on testnet from the selected snapshot to be certain that it would restart properly.

Unfortunately, it did not restart properly.

Why didn't it restart properly?

Because there was a verifier node on testnet that we did not restart. When testnet validator nodes came up, they connected to this verifier node and attempted to replay all the blocks between 9479 to 9574.

We were worried about this case because other verifiers run nodes we do not control, and we were afraid that they would also try to replay these extra blocks.

How was it solved?

By changing the port used for peer-to-peer connections on the blockchain. By using a different port, any external nodes that had not updated their software would no longer be able to connect to the ndau blockchain.

After changing the port, testnet came up properly and began processing blocks.

At this point (9:40 PM EDT), we felt we were ready to restart mainnet, and we did. It was up and running properly again at 10:15 PM EDT.

Lessons Learned

Don't roll back if there is any other option

Even though we only rolled back empty blocks, it was surprising and concerning for our customers and partners. ndau and the Tendermint blockchain offer immediate finality: once a block is committed it cannot be changed. Since ndev is currently operating all validator nodes we were able to perform this rollback and come back online quickly, but this violated our stated commitment to finality. We now have a better way to deal with this situation should it ever occur again.

Communicate with other node operators and partners

We attempted to fix the problem quickly ourselves to restore operation. We should have communicated and presented our plan to other node operators (including exchanges) before implementing it. This would have kept the blockchain offline much longer but node operators would not have been surprised.

Determinism can be challenging

EAI calculations are extremely complex. We specifically coded the `CreditEAI` transaction with extra attention to determinism, realizing that it might be an issue. However, we failed to test for the following combination of circumstances:

- unpredictable map iteration in the Go programming language
- integer division of small numbers
- multiple updates of a single account within a single transaction

We now have tests that verify this was the root cause and code that mitigates it.