With the recent tombstonings and jailings that happened on dYdX v4, I wanted to open a discussion around proper monitoring, security, and migration of dYdX validators.

The greater cosmos ecosystem has a flourishing community that creates and maintains tooling around monitoring their validators, and I'll get to those, but I first want to talk about best practices for validators. There's a huge amount to be said here, but I'm going to focus on 2 key points:

1. securing your priv_validator_key.json
2. securing your priv_validator_state.json

What is the priv_validator_key.json

location: ~/.dydxprotocol/config/priv_validator_key.json

This IS your validator. If there's one thing I want you to take away from this post, it's to backup your key, and never let it exist on more than 1 node at a time

. Tombstoning (permanent jailing) happens when this key is used on multiple nodes at the same time, by accident or not.

What is the priv_validator_state.json

location: ~/.dydxprotocol/data/priv_validator_state.json

This is the latest sign information of your validator. You want to protect this. If you're migrating your validator, it's not a bad idea to move this along side your priv_validator_key.json

in order to be absolutely certain you don't double sign.

How to migrate safely

Steps to migrate your validator safely:

1. let your current validator continue running
2. sync a 2nd node to current
3. stop your current validator (note that you'll start missing blocks here - THAT'S OKAY!)
4. move the priv_validator_key.json
5. restart the 2nd node (confirm your validator has started signing)
6. delete the priv_validator_key.json

from your old validator

— if you're intending to decommission your old dYdX node completely, you're done! —

1. if you intend to use the old validator as a backup, restart it

How to keep your validator safe

Use a remote signing solution. There are 2 options here:

- TMKMS
- Horcrux

These are critical because they separate the priv_validator_key.json

and priv_validator_state.json

from the node itself, allowing you to migrate without worrying about double signing. The primary difference between the two solutions is that:

- TMKMS uses a single remote signer to talk to a single node. TMKMS supports HSM to protect your key such as the YubiHSM.
- Horcrux is a many:many solution; you can have many signers point to many nodes, where the signers act like a

multisig. Horcrux does NOT support hardware key protection.

How to monitor your nodes

There are plenty of tools to monitor your signing, but Lavender.Five's preferred method isTenderduty. It's very easy to set up, and can notify you via Pagerduty, Slack, Telegram, Etc. if your node isn't signing.