

Cosmovisor

cosmovisor is a small process manager for Cosmos SDK application binaries that monitors the governance module for incoming chain upgrade proposals. If it sees a proposal that gets approved, cosmovisor can automatically download the new binary, stop the current binary, switch from the old binary to the new one, and finally restart the node with the new binary.

We recommend validators to use cosmovisor to run their nodes. This will make low-downtime upgrades smoother, as validators don't have to manually upgrade binaries during the upgrade. Instead, they can pre-install new binaries, and cosmovisor will automatically update them based on the on-chain software upgrade proposals.

Configuration

When Cosmovisor activates an upgrade, it does a backup of the entire data directory by default. This backup can take a very long time to process unless the user does aggressive historical-state-pruning using the pruning [configuration on the node](#).

As long as you have access to a previous state [snapshot](#), we recommend setting the environment variable `UNSAFE_SKIP_BACKUP` to `false` which skips the data backup and allows a much faster upgrade. If your node is configured to only keep a small amount of historical state, then you may be able to get away with running the backup quickly.

More information about Cosmovisor settings can be found in the [Cosmovisor documentation \(opens in a new tab\)](#).

Installation

Using go install

To install the latest version of cosmovisor, run the following command:

```
go install
```

```
cosmosdk.io/tools/cosmovisor/cmd/cosmovisor@latest
```

Manual Build

You can also install from source by pulling the cosmos-sdk repository and switching to the correct version and building as follows:

```
git clone
```

```
https://github.com/cosmos/cosmos-sdk.git cd
```

```
cosmos-sdk git checkout
```

`cosmovisor/vx.x.x` make cosmovisor This will build Cosmovisor in `/cosmovisor` directory. Afterwards you may want to put it into your machine's PATH like as follows:

```
cp cosmovisor/cosmovisor
```

```
~/go/bin/cosmovisor To check your Cosmovisor version, run
```

```
cosmovisor version
```

Directory structure

```
. |—— current -> genesis or upgrades/ |—— genesis | |—— bin | |—— DAEMON_NAME |—— upgrades |——  
|—— bin | |—— DAEMON_NAME |—— upgrade-info.json
```

Initializing Cosmovisor

1. Rename binary to `dydxprotocold`

```
mv dydxprotocold. <version>
```

```
- <platform>
```

`dydxprotocold` 1. Set the environment variables

```
export DAEMON_NAME = dydxprotocold export DAEMON_HOME =< your directory
```

1. The directory structure can be initialized with

```
cosmovisor init
```

```
< path
```

```
to
```

```
executable
```

- DAEMON_HOME
- should be set to the validator's home directory
- since Cosmovisor polls/data/
- for upgrade info.
- DAEMON_NAME
- should be set to dydxprotocold

How to run

cosmovisor is simply a thin wrapper around Cosmos applications. Use the following command to start a testnet validator using cosmovisor .

```
cosmovisor run
```

```
arg1
```

```
arg2
```

```
arg3
```

... All arguments passed to cosmovisor run will be passed to the application binary (as a subprocess). cosmovisor will return /dev/stdout and /dev/stderr of the subprocess as its own.

Example:

```
cosmovisor run
```

```
start
```

```
—log-level
```

```
info
```

```
—home
```

```
/dydxprotocol/chain/.alice runs
```

```
dydxprotocold start
```

```
—log-level
```

```
info
```

```
—home
```

```
/dydxprotocol/chain/.alice as its subprocess.
```

Last updated on May 29, 2024 [Performing Upgrades Using Cosmovisor to stage dYdX Chain binary upgrade](#)