# Cosmos Hub 3 Upgrade Proposal A

Figment Networks
July 26, 2019

## Context

Cosmos SDK is software that all validators must run in order to participate in the Cosmos Hub blockchain. The Tendermint team, All in Bits, has upgraded the Gaia-13004 testnet, the changes for which may be found in this blog post and more comprehensively in this changelog. The release is on track to be deployed on mainnet as cosmoshub-3. Some of the noteworthy changes include the items signalled in previous, successful governance proposals:

**1. Activate the Community Pool** - enable governance to spend funds from the community pool (full proposal)
**2. Don't Burn Deposits for Rejected Governance Proposals Unless Vetoed** - if a proposal gets rejected without being vetoed, the deposits will be returned to the depositors (and if a proposal fails to meet quorum, its deposit will still be burned) (full proposal)
**3. Notification for Security Critical Hard Fork at Block 482100** - critical security vulnerability in the codebase for the Cosmos Hub (additional info)
**4. Increase Max Validator Set Size to 125** - (full proposal)

The Cosmos SDK will also enable creating and voting on governance proposals that modify on-chain parameters without halting or forking the network, as well as the spending of community funds.

This is a proposal to approve these high-level changes for a final vote for what will become Cosmos Hub 3. Please read them carefully:
https://github.com/cosmos/cosmos-sdk/blob/rc1/v0.36.0/CHANGELOG.md

If approved, and assuming that testing is successful, there will be a second proposal called *Cosmos Hub 3 Upgrade Proposal B*. Cosmos Hub 3 Upgrade Proposal B should specify **1)** the software hash; **2)** the block height state export from cosmoshub-2; **3)** the genesis time; **4)** instructions for generating the new genesis file.

## Decision

We are signalling our approval to:

1. Accept this list of high-level changes: https://raw.githubusercontent.com/cosmos/cosmos-sdk/rc1/v0.36.0/CHANGELOG.md
2. Proceed to a final vote to signal the cosmoshub-3 upgrade based upon these changes.

## Justification

These upgrades reflect the preferences signalled by the voters in prior governance proposal votes. These upgrades advance the Cosmos Hub's functionality and intended value.

In no particular order, the stakeholder parties likely to be impacted by advancing this decision:

1. Software developers
2. Validators
3. Delegators

This decision will likely impact the stakeholders in these ways:

1. Software developers will have the burden of releasing new software that is compatible with the breaking changes.
2. Validators and Delegators will have the burden of verifying and/or accepting the new Cosmos SDK software code prior to the second vote.
3. Validators will have the burden of voting on a second upgrade proposal.
4. Delegators may have the burden of voting on two Cosmos SDK software upgrade proposals.

This proposal is only to signal acceptance/rejection of the changes that are proposed to be included in the Cosmos Hub 3 upgrade at a high level, and in preparation for a second vote upon the precise software that will be released. It should always be each validator's responsibility to decide what software they will run, and nothing in this proposal should be construed to undermine that responsibility.

Notes and/or Future Considerations

1. A Cosmos Hub 3 Upgrade Proposal B will follow.
2. As stakeholders, we will want upgrades to improve the Cosmos SDK software via the governance process. Thus, we should establish a formal process by which various parties may contribute upgrades to the Cosmos Hub.

# Changelog

## 0.36.0

### Breaking Changes

- #3565 Updates to the governance module:
    - Rename JSON field from `proposal_content` to `content`
    - Rename JSON field from `proposal_id` to `id`
    - Disable `ProposalTypeSoftwareUpgrade` temporarily
- #3775 unify sender transaction tag for ease of querying
- #4255 Add supply module that passively tracks the supplies of a chain
    - Renamed `x/distribution` `ModuleName`
    - Genesis JSON and CLI now use `distribution` instead of `distr`
    - Introduce `ModuleAccount` type, which tracks the flow of coins held within a module
    - Replaced `FeeCollectorKeeper` for a `ModuleAccount`
    - Replaced the staking `Pool`, which coins are now held by the `BondedPool` and `NotBonded` module accounts
    - The `NotBonded` module account now only keeps track of the not bonded tokens within staking, instead of the whole chain
    - #3628 Replaced governance's burn and deposit accounts for a `ModuleAccount`
    - Added a `ModuleAccount` for the distribution module
    - Added a `ModuleAccount` for the mint module #4472 validation for crisis genesis
- #3985 `ValidatorPowerRank` uses potential consensus power instead of tendermint power
- #4104 Gaia has been moved to its own repository: https://github.com/cosmos/gaia

- #4104 Rename gaiad.toml to app.toml. The internal contents of the application config remain unchanged.
- #4159 create the default module patterns and module manager
- #4230 Change the type of ABCIMessageLog#MsgIndex to uint16 for proper serialization.
- #4250 BaseApp.Query() returns app's version string set via BaseApp.SetAppVersion() when handling /app/version queries instead of the version string passed as build flag at compile time.
- #4262 GoSumHash is no longer returned by the version command.
- #4263 RestServer#Start now takes read and write timeout arguments.
- #4305 `GenerateOrBroadcastMsgs` no longer takes an `offline` parameter.
- #4342 Upgrade go-amino to v0.15.0
- #4351 InitCmd, AddGenesisAccountCmd, and CollectGenTxsCmd take node's and client's default home directories as arguments.
- #4387 Refactor the usage of tags (now called events) to reflect the new ABCI events semantics:
    - Move `x/{module}/tags/tags.go` => `x/{module}/types/events.go`
    - Update `docs/specs`
    - Refactor tags in favor of new `Event(s)` type(s)
    - Update `Context` to use new `EventManager`
    - (Begin|End)Blocker no longer return tags, but rather uses new `EventManager`
    - Message handlers no longer return tags, but rather uses new `EventManager` Any component (e.g. BeginBlocker, message handler, etc...) wishing to emit an event must do so through `ctx.EventManger().EmitEvent(s)`. To reset or wipe emitted events: `ctx = ctx.WithEventManager(sdk.NewEventManager())` To get all emitted events: `events := ctx.EventManager().Events()`
- #4437 Replace governance module store keys to use `[]byte` instead of `string`.
- #4451 Improve modularization of clients and modules:
    - Module directory structure improved and standardized
    - Aliases autogenerated
    - Auth and bank related commands are now mounted under the respective moduels
    - Client initialization and mounting standardized
- #4479 Remove codec argument redundency in client usage where the CLIContext's codec should be used instead.
- #4488 Decouple client tx, REST, and ultil packages from auth. These packages have been restructured and retrofitted into the `x/auth` module.

- #4521 Flatten x/bank structure by hiding module internals.
- #4525 Remove --cors flag, the feature is long gone.
- #4536 The `/auth/accounts/{address}` now returns a `height` in the response. The account is now nested under `account`.
- #4543 Account getters are no longer part of client.CLIContext() and have now moved to reside in the auth-specific AccountRetriever.
- #4588 Context does not depend on x/auth anymore. client/context is stripped out of the following features:
  - GetAccountDecoder()
  - CLIContext.WithAccountDecoder()
  - CLIContext.WithAccountStore() x/auth.AccountDecoder is unnecessary and consequently removed.
- #4602 client/input.{Buffer,Override}Stdin() functions are removed. Thanks to cobra's new release they are now redundant.
- #4633 Update old Tx search by tags APIs to use new Events nomenclature.
- #4649 Refactor x/crisis as per modules new specs.
- #3685 The default signature verification gas logic (`DefaultSigVerificationGasConsumer`) now specifies explicit key types rather than string pattern matching. This means that zones that depended on string matching to allow other keys will need to write a custom `SignatureVerificationGasConsumer` function.
- #4663 Refactor bank keeper by removing private functions
  - `InputOutputCoins`, `SetCoins`, `SubtractCoins` and `AddCoins` are now part of the `SendKeeper` instead of the `Keeper` interface
- (tendermint) #4721 Upgrade Tendermint to v0.32.1

## Features

- #2020 New keys export/import command line utilities to export/import private keys in ASCII format that rely on Keybase's new underlying ExportPrivKey()/ImportPrivKey() API calls.
- #3565 Implement parameter change proposal support. Parameter change proposals can be submitted through the CLI or a REST endpoint. See docs for further usage.
- #3850 Add `rewards` and `commission` to distribution tx tags.
- #3981 Add support to gracefully halt a node at a given height via the node's `halt-height` config or CLI value.
- #4144 Allow for configurable BIP44 HD path and coin type.
- #4250 New BaseApp.{,Set}AppVersion() methods to get/set app's version string.

- #4263 Add `--read-timeout` and `--write-timeout` args to the `rest-server` command to support custom RPC R/W timeouts.
- #4271 Implement Coins#IsAnyGT
- #4318 Support height queries. Queries against nodes that have the queried height pruned will return an error.
- #4409 Implement a command that migrates exported state from one version to the next. The `migrate` command currently supports migrating from v0.34 to v0.36 by implementing necessary types for both versions.
- #4570 Move /bank/balances/{address} REST handler to x/bank/client/rest. The exposed interface is unchanged.
- Community pool spend proposal per Cosmos Hub governance proposal #7 "Activate the Community Pool"

## Improvements

- #2286 Improve performance of CacheKVStore iterator.
- #3512 Implement Logger method on each module's keeper.
- #3655 Improve signature verification failure error message.
- #3774 add category tag to transactions for ease of filtering
- #3914 Implement invariant benchmarks and add target to makefile.
- #3928 remove staking references from types package
- #3978 Return ErrUnknownRequest in message handlers for unknown or invalid routed messages.
- #4190 Client responses that return (re)delegation(s) now return balances instead of shares.
- #4194 ValidatorSigningInfo now includes the validator's consensus address.
- #4235 Add parameter change proposal messages to simulation.
- #4235 Update the minting module params to implement params.ParamSet so individual keys can be set via proposals instead of passing a struct.
- #4259 `Coins` that are `nil` are now JSON encoded as an empty array `[]`. Decoding remains unchanged and behavior is left intact.
- #4305 The `--generate-only` CLI flag fully respects offline tx processing.
- #4379 close db write batch.
- #4384- Allow splitting withdrawal transaction in several chunks
- #4403 Allow for parameter change proposals to supply only desired fields to be updated in objects instead of the entire object (only applies to values that are objects).
- #4415 /client refactor, reduce genutil dependancy on staking
- #4439 Implement governance module iterators.

- #4465 Unknown subcommands print relevant error message
- #4466 Commission validation added to validate basic of MsgCreateValidator by changing CommissionMsg to CommissionRates
- #4501 Support height queriers in rest client
- #4535 Improve import-export simulation errors by decoding the `KVPair.Value` into its respective type
- #4536 cli context queries return query height and accounts are returned with query height
- #4553 undelegate max entries check first
- #4556 Added IsValid function to Coin
- #4564 client/input.GetConfirmation()'s default is changed to No.
- #4573 Returns height in response for query endpoints.
- #4580 Update `Context#BlockHeight` to properly set the block height via `WithBlockHeader`.
- #4584 Update bank Keeper to use expected keeper interface of the AccountKeeper.
- #4584 Move `Account` and `VestingAccount` interface types to `x/auth/exported`.
- #4082 supply module queriers for CLI and REST endpoints
- #4601 Implement generic pangination helper function to be used in REST handlers and queriers.
- #4629 Added warning event that gets emitted if validator misses a block.
- #4674 Export `Simapp` genState generators and util functions by making them public
- #4706 Simplify context Replace complex Context construct with a simpler immutible struct. Only breaking change is not to support `Value` and `GetValue` as first class calls. We do embed ctx.Context() as a raw context.Context instead to be used as you see fit.
  Migration guide:
  `ctx = ctx.WithValue(contextKeyBadProposal, false)` -> `ctx = ctx.WithContext(context.WithValue(ctx.Context(), contextKeyBadProposal, false))`
  A bit more verbose, but also allows `context.WithTimeout()`, etc and only used in one function in this repo, in test code.
- #3685 Add `SetAddressVerifier` and `GetAddressVerifier` to `sdk.Config` to allow SDK users to configure custom address format verification logic (to override the default limitation of 20-byte addresses).
- #3685 Add an additional parameter to NewAnteHandler for a custom `SignatureVerificationGasConsumer` (the default logic is now in `DefaultSigVerificationGasConsumer). This allows SDK users to configure their

own logic for which key types are accepted and how those key types consume gas.
- Remove `--print-response` flag as it is no longer used.
- Revert #2284 to allow create_empty_blocks in the config
- (tendermint) #4718 Upgrade tendermint/iavl to v0.12.3

## Bug Fixes

- #1351 Stable AppHash allows no_empty_blocks
- #3705 Return `[]` instead of `null` when querying delegator rewards.
- #3966 fixed multiple assigns to action tags #3793 add delegator tag for MsgCreateValidator and deleted unused moniker and identity tags
- #4194 Fix pagination and results returned from /slashing/signing_infos
- #4230 Properly set and display the message index through the TxResponse.
- #4234 Allow `tx send --generate-only` to actually work offline.
- #4271 Fix addGenesisAccount by using Coins#IsAnyGT for vesting amount validation.
- #4273 Fix usage of AppendTags in x/staking/handler.go
- #4303 Fix NewCoins() underlying function for duplicate coins detection.
- #4307 Don't pass height to RPC calls as Tendermint will automatically use the latest height.
- #4362 simulation setup bugfix for multisim 7601778
- #4383 - currentStakeRoundUp is now always atleast currentStake + smallest-decimal-precision
- #4394 Fix signature count check to use the TxSigLimit param instead of a default.
- #4455 Use `QueryWithData()` to query unbonding delegations.
- #4493 Fix validator-outstanding-rewards command. It now takes as an argument a validator address.
- #4598 Fix redelegation and undelegation txs that were not checking for the correct bond denomination.
- #4619 Close iterators in `GetAllMatureValidatorQueue` and `UnbondAllMatureValidatorQueue` methods.
- #4654 validator slash event stored by period and height
- #4681 panic on invalid amount on `MintCoins` and `BurnCoins`
    - skip minting if inflation is set to zero

## 0.35.0

### Bug Fixes

- Fix gas consumption bug in `Undelegate` preventing the ability to sync from genesis.

## 0.34.7

### Bug Fixes

#### SDK

- Fix gas consumption bug in `Undelegate` preventing the ability to sync from genesis.

## 0.34.6

### Bug Fixes

#### SDK

- Unbonding from a validator is now only considered "complete" after the full unbonding period has elapsed regardless of the validator's status.