### Moving on forward

* Internal Informal audit repository will be created for osmosis artifacts -> report issues on osmosis main, with Informal tag
* Audit will be continued on osmosis main commit hash: [6fcd25f](https://github.com/osmosis-labs/cosmos-sdk/commit/6fcd25f8abb82ac4db337d72510429568362d348)
* Mirel’s vacation period from 12.08. until 28.08. - Aleksandar who is currently onboarding will work part time on the osmosis audit project and will continue with analysis of the possible osmosis modules impact. -> tag Dev
* All the work will be overviewed in the week after the vacation -> suggestion that the next sync meeting is moved from: 25.08. to 01.09.
* Estimation of completed analysis of before send hook: 08.09.
* After that ForceTransfer and Burn features can be analyzed.

### Work done so far

**Possible impact analysis**

Impact of added Before Send hook overview: diagrams created - Core and osmosis modules need to be analyzed - we organized analysis in two tasks/phases.

* **Core Cosmos SDK analysis**
* Diagrams for each of the impact paths created. Will be shared.
* Conclusions made when taken into account: differences/similarities to bank’s blacklisted feature:
* The Vanilla Bank module already has a feature that is somewhat similar to the implemented freezing of Coins transfer. Accounts (not Module accounts) can be blacklisted, and transferring from/to accounts can be “freezed” in the bank module. The error messages containing info are propagated and the call stack is taking into account that sending can be failing due to this case. (Not only insufficient funds, or missing module accounts.)
* Code inspection is done in order to analyze call stack for modules
* Business logic taken into account and concluded that for:
  + Staking, Vesting/Clawback vesting: modules will not work with native tokens created with Token Factory.
  + Crisis Module: ConstantFee is paid in OSMO tokens, so this should not be an issue during VerifyInvariant, where the Sending is used.
  + Governance Impact: proposals are deposited in OSMO tokens.

#### **Distribution module impact** in BeginBlocker -> AllocateTokens

On the kick off it was stated that :

* if transferring one type of coins fails, we should continue with sending

Implemented functions have different logic:

* Before Send hook is implemented in a way that if for one denom sending is freezed, hook will return an error, and freeze the sending.
* SendManyCoins, SendCoins… - will fail if one of the pairs recipient-amount fails

If the different behavior is allowed in the distribution module, this will this introduce behavior that is not clearly distincted?

Begin blocker: <https://github.com/osmosis-labs/cosmos-sdk/blob/a296bdc79450be41666e1d235ffd5ed0269305dd/x/distribution/abci.go#L30-L33>

Allocation of tokens: sending fees from fee collector module acc -> distribution module acc

<https://github.com/osmosis-labs/cosmos-sdk/blob/a296bdc79450be41666e1d235ffd5ed0269305dd/x/distribution/keeper/allocation.go#L30-L34>

Questions:

* What is the expected behavior in other modules?
* Just to confirm: it is intended to give a possibility to blocklist distribution module accounts (or any other module account) to a token admin?
  + Meeting MoMs:
  + Module accounts should not be blocked, this will be a feature. This will make their accounting more complicated?- will this be implemented?
  + Distribution works only with OSMOs.
  + Is the distribution module allowed to receive coins? Check this!!!
  + Swaps?

If not -> maybe hooks should not be triggered for SendCoinsFromModuleToModule() => different send function when sending from/to modules?

**1.b Distribution Module -> FundCommunityPool()**

Is called from several osmosis modules. My understanding is that community pools could be containing all different kinds of denoms, including token factory native denoms.

* **Pool Incentives** is using this function from:

Implementation of Mint Hook AfterDistributeMintedCoin →AllocateAsset → FundCommunityPoolFromModule

* **Mint Module** is calling this function from  
  Implementation of AfterEpochsEnd hook → DistributeMintedCoins

!!! Both hooks will panic if called functions return an error (which could happen with the FundCommunityPool function)

Different approach - bypassing the hook for module accounts?

Introducing new hooks - from/to hooks?

TODO for Informal audit crew: Continue with analysis of osmosis modules, to find problematic places.

#### **CosmWasm smart contract** CW20 standard

SC will implement BeforeSend logic - based on which, we will have the hook returning an error in Sending functions execution and freeze them:

If the hook is registered for one native denom - smart contract logic (behind the registered address) should work with only one native token.

Sudo execution from BeforeSend hook implementation is called per denom.

**types.SudoMsg creation:**

<https://github.com/osmosis-labs/osmosis/blob/f09305e60b5b23fec761ea14446ee33556313e69/x/tokenfactory/keeper/beforesend.go#L90-L96>

#### IBC impact

* Regarding impact on IBC - ICA: there is no concern that before sending hook triggering will introduce large issues - since only packages are transferred, and once the package is received on the host side, a message will be executed.

Once the package is received on the osmosis side, the sending message will be executed and if sending is freezed, error will be propagated.

NewErrorAcknowledgment will be set - as for any other message that could occur.

If the execution is successful -> marshaled message response will be returned.

<https://github.com/cosmos/ibc-go/blob/6c3420e1a0c0eb67120635616ac545580c22483e/modules/apps/27-interchain-accounts/host/keeper/relay.go#L31-L36>

* Regarding transfer app and osmosis integration with it - remains to be analyzed
* Informal’s IBC experts will be consulted to confirm this analysis.

#### **Error handling** (not that important)

defining sentinel errors for each of the modules (Bank, TokenFactory) could be defined for better error handling.

For bank module:

ErrFrozenSendingOfCoins = sdkerrors.Register(ModuleName, N, "sending freezed due to triggered before sending hook)

For TokenFactory module:

ErrSudoExecutionOfBeforeSendMsgFailed = sdkerrors.Register(ModuleName, **N**, "sending freezed due to triggered before sending hook