Icacallbacks

Suggest Edits

The ICACallbacks Module

Add icacallbacks module. Interchain accounts are very useful, but ICA calls triggered by automated logic on Stride are limited in functionality and difficult to work with due to a lack of callbacks. Most of Stride's interchain account logic is triggered epochly from the BeginBlocker , and state updates on Stride must be made after ICA calls are issued, based on the success / failure of those calls.

The challenges faced before creating the icacallbacks modules were: (1) Messages must be handled one-off (by matching on message type) - really what we want to do is update state in acks based on thetransaction sent (2) Message responses are almost always empty, e.g. MsgDelegateResponse , which leads to (3) Matching processed messages to state on the controller is very challenging (sometimes impossible) based on the attributes of the message sent. For example, given the following type

type Gift struct { from string to string amount int reason string } two ICA bank sends associated with gifts that have the same from , to and amount butdifferent

reasons are indistinguishable today in acks.

icacontroller solves the issues as follows

- Callbacks can be registered to process data associated with a particular IBC packet, per module (solves 1)
- ICA auth modules can store callback data using
- icacallbacks
- , passing in both a callback and args
- Arguments to the callback can be (un)marshaled and contain arbitrary keys, allowing for updates to data on the controller chain based on the success / failure of the ICA call (solves 2 / 3)

Technical notes

- Callbacks are uniquely identifiable through
- portId/channelId/sequence
- keys
- Only modules that have registered callbacks can invoke them
- icacallbacks
- doesn't have a message server / handler (it can only be called by other modules)
- icacallbacks
- does authentication by fetching the module associated with a packet (containing the registered callbacks) by calling
- ChannelKeeper.LookupModuleByChannel
- (it's permissioned at the module level)
- icacallbacks
- is an interchain account auth module, although it's possible this design could be generalized to work with other IBC modules
- in case of a timeout, callbacks are still executed with the ack set to an empty byte array
- We're using protos to serialize / deserialize callback arguments

The flow to add callbacks is to call ICACallbacksKeeper.SetCallbackData after sending an IBC transaction. When the ack returns

- the callback is fetched using the callback key
- the module is fetched using the portId / channelId
- and the callback is invoked and deleted.

The middleware structure is as follows

Invariants

- portId, channelId
- pair map to a unique module (important for fetching the correct
- CallbackHandler
- from a received packet)
- callback ids are unique within modules (they don't have to be unique between modules, because
- CallICACallback
- is scoped to a module)

Keeper functions

- CallRegisteredICACallback()
- : invokes the relevant callback associated with an ICA

State

- CallbackData
- : stores the callback type, arguments and associated packet
- CallbackHandler
- Callbacks
- Callback

Events

The icacallbacks module does not currently emit any events. Updated3 months ago

Claim Interchainquery Did this page help you?Yes No *Table of Contents * *The ICACallbacks Module * * *Keeper functions * * *State * * *Events