[](https://ida.interchain.io/)

[Interchain Developer Academy](https://ida.interchain.io/)/[Interchain Developer Academy](https://ida.interchain.io/tutorials/6-ibc-dev/)



Search

[Interchain Developer Academy](https://ida.interchain.io/)[Interchain Developer Academy](https://ida.interchain.io/tutorials/6-ibc-dev/)

Search



Filters

Interchain Developer Academy

[](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Week 0 - Getting Started](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Getting Started](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Blockchain 101](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Blockchain History](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Public and Managed Blockchains](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Consensus in Distributed Networks](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Cryptography](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Self-Assessment Quiz](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Go Introduction - First Steps](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Go Basics](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Go Interfaces](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Control Structures in Go](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Arrays and Slices in Go](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Standard Packages in Go](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Concurrency in Go](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Good-To-Know Dev Terms](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Docker Introduction](https://ida.interchain.io/tutorials/6-ibc-dev/)

[](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Week 1 - Introduction to the Interchain](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Introduction to the Interchain](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Blockchain Technology and the Interchain](https://ida.interchain.io/tutorials/6-ibc-dev/)

[The Interchain Ecosystem](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Getting ATOM and Staking It](https://ida.interchain.io/tutorials/6-ibc-dev/)

[A Blockchain App Architecture](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Accounts](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Transactions](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Messages](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Modules](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Protobuf](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Multistore and Keepers](https://ida.interchain.io/tutorials/6-ibc-dev/)

[BaseApp](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Queries](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Events](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Context](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Testing](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Relaying with IBC](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Interchain Security](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Bridges](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Migrations](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Week 1 Quiz](https://ida.interchain.io/tutorials/6-ibc-dev/)

[](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Week 2 - First Steps](https://ida.interchain.io/tutorials/6-ibc-dev/)

[First Steps](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Setup Your Work Environment](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Run a Node, API, and CLI](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Ignite CLI](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Exercise - Make a Checkers Blockchain](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Store Object](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Create Custom Messages](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Create and Save a Game Properly](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Add a Way to Make a Move](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Emit Game Information](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Record the Game Winner](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Week 2 Exercise](https://ida.interchain.io/tutorials/6-ibc-dev/)

[](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Week 3 - Introduction to IBC and CosmJS](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Introduction to IBC and CosmJS](https://ida.interchain.io/tutorials/6-ibc-dev/)

[What is IBC?](https://ida.interchain.io/tutorials/6-ibc-dev/)

[IBC/TAO - Connections (OPTIONAL)](https://ida.interchain.io/tutorials/6-ibc-dev/)

[IBC/TAO - Channels (OPTIONAL)](https://ida.interchain.io/tutorials/6-ibc-dev/)

[IBC/TAO - Clients (OPTIONAL)](https://ida.interchain.io/tutorials/6-ibc-dev/)

[IBC Token Transfer](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Interchain Accounts (OPTIONAL)](https://ida.interchain.io/tutorials/6-ibc-dev/)

[IBC Middleware (OPTIONAL)](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Create a Custom IBC Middleware (OPTIONAL)](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Integrate IBC Middleware Into a Chain (OPTIONAL)](https://ida.interchain.io/tutorials/6-ibc-dev/)

[IBC Tooling](https://ida.interchain.io/tutorials/6-ibc-dev/)

[What is CosmJS?](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Your First CosmJS Actions](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Compose Complex Transactions](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Learn to Integrate Keplr](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Create Custom CosmJS Interfaces](https://ida.interchain.io/tutorials/6-ibc-dev/)

[](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Week 4 - Ignite CLI and IBC Advanced](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Ignite CLI and IBC Advanced](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Keep an Up-To-Date Game Deadline](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Keep Track Of How Many Moves Have Been Played](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Put Your Games in Order](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Auto-Expiring Games](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Let Players Set a Wager](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Handle wager payments](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Integration tests](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Incentivize Players](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Help Find a Correct Move](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Play With Cross-Chain Tokens](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Understand IBC Denoms](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Go Relayer](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Hermes Relayer](https://ida.interchain.io/tutorials/6-ibc-dev/)

[](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Week 5 - CosmJS Advanced](https://ida.interchain.io/tutorials/6-ibc-dev/)

[CosmJS Advanced](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Create Custom Objects](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Create Custom Messages](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Get an External GUI](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Integrate CosmJS and Keplr](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Backend Script for Game Indexing](https://ida.interchain.io/tutorials/6-ibc-dev/)

[](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Week 6 - IBC Deep Dive](https://ida.interchain.io/tutorials/6-ibc-dev/)

[IBC Deep Dive](https://ida.interchain.io/tutorials/6-ibc-dev/)

[IBC Application Developer Introduction](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Make a Module IBC-Enabled](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Adding Packet and Acknowledgment Data](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Extend the Checkers Game With a Leaderboard](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Create a Leaderboard Chain](https://ida.interchain.io/tutorials/6-ibc-dev/)

[](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Week 7 - From Code to MVP to Production and Migrations](https://ida.interchain.io/tutorials/6-ibc-dev/)

[From Code to MVP to Production and Migrations](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Run in Production](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Prepare the Software to Run](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Prepare a Validator and Keys](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Prepare Where the Node Starts](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Prepare and Connect to Other Nodes](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Configure, Run, and Set Up a Service](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Prepare and Do Migrations](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Simulate Production in Docker](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Tally Player Info After Production](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Add a Leaderboard as a Module](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Migrate the Leaderboard Module After Production](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Simulate a Migration in Docker](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Final Exam](https://ida.interchain.io/tutorials/6-ibc-dev/)

[](https://ida.interchain.io/tutorials/6-ibc-dev/)

[What's Next?](https://ida.interchain.io/tutorials/6-ibc-dev/)

[Continue Your Interchain Journey](https://ida.interchain.io/tutorials/6-ibc-dev/)

Docs Version Switcher

On this page

[ICS-20 - token transfer](https://ida.interchain.io/tutorials/6-ibc-dev/#ics-20-token-transfer)

[How are IBC denoms derived?](https://ida.interchain.io/tutorials/6-ibc-dev/#how-are-ibc-denoms-derived)

[Practical example: denom\_trace](https://ida.interchain.io/tutorials/6-ibc-dev/#practical-example-denom-trace)

[The chain ID and the client ID](https://ida.interchain.io/tutorials/6-ibc-dev/#the-chain-id-and-the-client-id)

[Ensure the IBC client is not expired](https://ida.interchain.io/tutorials/6-ibc-dev/#ensure-the-ibc-client-is-not-expired)

[🎉Congratulations!🎉](https://ida.interchain.io/tutorials/6-ibc-dev/#%F0%9F%8E%89congratulations-%F0%9F%8E%89)

[#Copy link](https://ida.interchain.io/tutorials/6-ibc-dev/#understand-ibc-denoms) **Understand IBC Denoms**

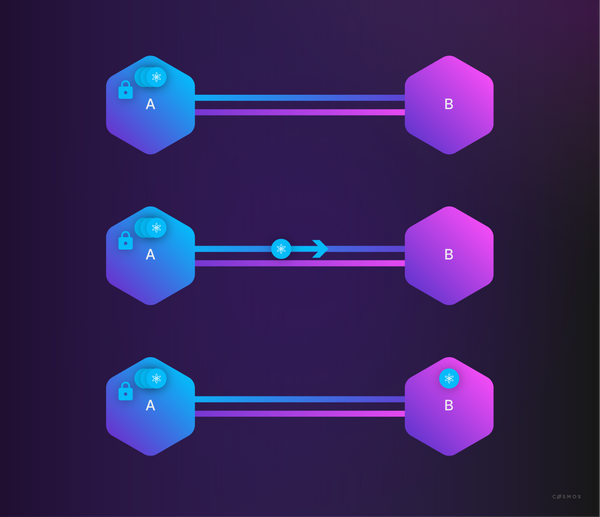
The Interchain vision set out by the original [Cosmos whitepaper (opens new window)↗](https://v1.cosmos.network/resources/whitepaper) was one of sovereign, application-specific, Proof-of-Stake blockchains. A crucial component of this vision was the **Inter-Blockchain Communication Protocol** or simply **IBC**. With IBC, chains can maintain their sovereignty while still being able to permissionlessly inter-operate with other chains (that also enable IBC), thus paving the way towards an *internet of blockchains*.

*Sounds great, right? But wait, what does that actually mean?*

Well, IBC enables arbitrary message passing between chains (in fact, even more generalized state machines like a [solo machine (opens new window)↗](https://interchain-io.medium.com/ibc-beyond-light-clients-solo-machine-5af93618ef63)), so developers can go ahead and create all sorts of IBC applications that exchange packets of data over IBC to enable application logic.

However, the first and still most dominant example to date is to transfer a (fungible) token from a source chain to a destination chain.

Take this example: you have some ATOM on the Cosmos Hub but would like to swap this for some other token on a DEX (**D**ecentralized **Ex**change) like [Osmosis (opens new window)↗](https://app.osmosis.zone/). This can be illustrated with a random IBC transfer between the Hub and Osmosis using Mintscan, a popular block explorer.



*Sending token from blockchain A to blockchain B*

Take the [following transaction (opens new window)↗](https://www.mintscan.io/cosmos/txs/F7196B37828BAAF5C55E499D62A58E2927542CB2FB57B587BA77BF5BB044FFBF). There you see some general information about the transaction, as well as data, particularly on the IBC transfer message that was included in the transaction. Dropping sender and receiver you find:

| **Key** | **Value** |
| --- | --- |
| Source Channel | channel-141 |
| Port | transfer |
| Sequence | 1,269,133 |
| Amount | 0.020000 ATOM |
| Origin Amount | 20,000 |
| Origin Denom | uatom |

If you are familiar with the [basics of IBC](https://ida.interchain.io/academy/3-ibc/1-what-is-ibc.html), you will know what to make of these terms.

Now, what if you want to send some ATOM back from Osmosis to the Hub? An example would be [this transaction (opens new window)↗](https://www.mintscan.io/osmosis/txs/9721FE816ABEE87D25259F87BA816EB53651194DD871C3F6B0A5B00434429A80)

| **Key** | **Value** |
| --- | --- |
| Source Channel | channel-0 |
| Port | transfer |
| Sequence | 1,265,787 |
| Amount | 1.850271 ATOM |
| Origin Amount | 1,850,271 |
| Origin Denom | ? |

Now instead of seeing uatom in the Origin Denom field, you see: IBC/27394FB092D2ECCD56123C74F36E4C1F926001CEADA9CA97EA622B25F41E5EB2.

This is what is called an **IBC denom**, and this is the way assets sent over IBC will be represented on-chain.



In this tutorial you will:

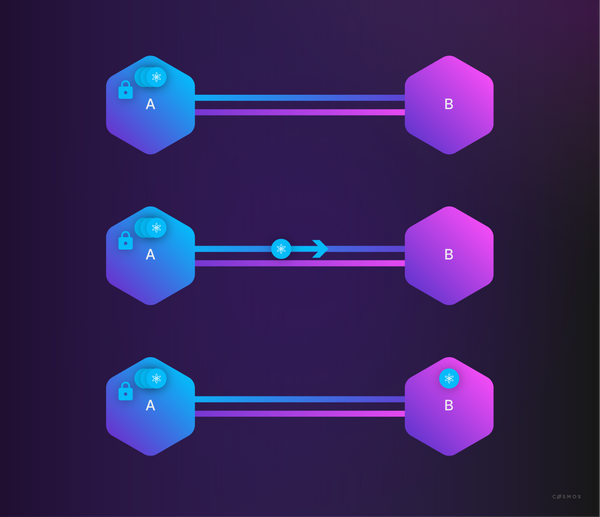
* Look at the context of ICS-20.
* Explain how the IBC denom is derived.
* Learn how to trace back the original denom.
* Find out what chain denoms are coming from.

[#Copy link](https://ida.interchain.io/tutorials/6-ibc-dev/#ics-20-token-transfer) ICS-20 - token transfer



Token transfers or ICS-20 is discussed in detail in the [IBC section](https://ida.interchain.io/academy/3-ibc/7-token-transfer.html). The "ICS" in ICS-20 is shorthand for *Interchain standards*. In the section, you can find an in-depth look at how IBC enables the transfer of (fungible) tokens across chains. For the purposes of this tutorial, here comes a brief and simplified summary.

Imagine two blockchains, blockchain A and blockchain B. As a starting point, you have some tokens on blockchain A you want to send to blockchain B. You can follow the steps in the image below:



*Sending tokens from blockchain A to blockchain B*

When sending the tokens to another blockchain with IBC:

1. A packet commitment is stored on blockchain A and the tokens to be sent are escrowed on chain A (top left of the image).
2. A relayer takes note of the packet to be sent and submits a MsgRecvPacket on the destination chain, along with a proof to be verified by the chain A light client on chain B (middle of the image).
3. **With IBC, the value that the tokens represent can be transferred across chains, but the token itself cannot.** Therefore, blockchain B mints its own representative tokens in the form of *voucher* replacement tokens. These will be characterized by the IBC denoms IBC/... (bottom right of the image).



Note that this only considers the *happy path* where the token transfer is successful and the respective proofs can be verified.

When sending the tokens back with IBC to the source blockchain:

1. Blockchain B sends the voucher tokens back to blockchain A.
2. The voucher tokens are destroyed (burned) on blockchain B.
3. The locked tokens on blockchain A are unlocked.



The only way to unlock the locked tokens on blockchain A is to send the voucher tokens back from blockchain B. The result is that the voucher tokens on blockchain B are burned. The burn process purposefully takes the vouchers out of circulation.

[#Copy link](https://ida.interchain.io/tutorials/6-ibc-dev/#how-are-ibc-denoms-derived) How are IBC denoms derived?

IBC is a protocol that allows for permissionless creation of clients, connections, and channels by relayers. Again, refer to the [IBC section](https://ida.interchain.io/academy/3-ibc/7-token-transfer.html) for more in-depth information. As explained there, **a consequence of the permissionless creation of clients, connections, and channels is that tokens that have traveled different paths have different security guarantees**. To account for this, the IBC protocol makes sure to prepend the path information to a base denomination when representing the vouchers minted on the sink chain when transferring tokens over IBC.



Taking once more the example of Osmosis and the Cosmos Hub, when sending some OSMO from Osmosis to the Hub the channelEnd on the Hub side is characterized by:



Copy

channelEnd{

...

counterpartyChannelIdentifier: 'channel-141',

counterpartyPortIdentifier: 'transfer'

...

}

This is prepended according to the format: {portID}/{channelID}/base\_denom

In this particular example: transfer/channel-141/uosmo.

This representation of an IBC asset (where the path information is prepended to the base denomination the asset is representing) is a valid representation of the asset and can be found on some user interfaces. However, it is not quite the previously cited IBC denom IBC/....

In fact, there is one more step required to arrive at the IBC denom. You take the hash of the base\_denom prepended with the path information, using the SHA256 hashing function. This gives the following for the IBC denom:



Copy

// hash() representing a SHA256 hashing function returning a string

ibc\_denom := 'ibc/' + hash('path' + 'base\_denom')



In the example from earlier, with transfer/channel-141/uosmo, the corresponding IBC denom is: ibc/14F9BC3E44B8A9C1BE1FB08980FAB87034C9905EF17CF2F5008FC085218811CC.   
  
Note that the assets transferred over IBC are stored on-chain as IBC denoms. It is however up to developers of frontends and user interfaces to decide whether they will use the human-readable form instead to fit their UX needs.

It is possible to use a query to find the hash based on the path information of the IBC asset, as will be described; however, you can always calculate it using a [SHA256 hash generator (opens new window)↗](https://xorbin.com/tools/sha256-hash-calculator) as well.



**So...why use a hash?**

Hashing functions have many desirable properties that make them often used in cryptography. The property most useful in this discussion is that the hashed output is always reduced to a fixed length (256 bits in the case of SHA256), no matter the length of the input.   
  
Consider the following:

* The hash could contain paths that track the token on multiple hops from chain to chain.
* This could potentially be unbearably long when directly printing the path.
* The Cosmos SDK has a 64-character limit on the denomination of the token.

This is why a hash was preferred. More information on the design decisions can be found [here (opens new window)↗](https://ibc.cosmos.network/main/architecture/adr-001-coin-source-tracing.html).

The trade-off when using a hash is that you cannot compute the input given the output (hashing is an irreversible operation). Therefore, the ICS-20 module keeps a mapping of IBC denominations it has encountered in order to look up the original path and base\_denom. Therefore, you are required to query a node to find out what the actual path and denomination are. This query is called the *denomtrace*.



With IBC denoms, there is a special meaning for the / character. It is used to parse port and channel identifiers from the base denom. Hence it was initially forbidden to use forward slashes in the base denomination of tokens.   
  
However, due to a requirement from the Evmos chain (which uses forward slashes in contract-based denoms), support for base denominations containing / has been added. For more information, check the [ibc-go documentation (opens new window)↗](https://ibc.cosmos.network/main/migrations/support-denoms-with-slashes.html).

[#Copy link](https://ida.interchain.io/tutorials/6-ibc-dev/#practical-example-denom-trace) Practical example: denom\_trace

You can distinguish two cases of interacting with IBC denoms:

1. Calculating the IBC denom for a given path.
2. Tracing the path information and base denom when encountering an IBC denom.

The transfer IBC module exposes queries for both of these cases. In order to query, you will have to interact with a node of the blockchain network.

These queries are:



Copy

1. $ <binary> query ibc-transfer denom-hash [trace] [flags]

2. $ <binary> query ibc-transfer denom-trace [hash] [flags]

This tutorial uses the gaiad binary from the Cosmos Hub, continuing with the previous example where some OSMO is transferred from Osmosis to the Hub.



Despite using gaiad as an example here, you should use the chain binary of *the chain where the asset you are interested in is present*. The transfer IBC module needs to look up the mapping it stores when querying *denom\_trace*.

Install the Gaia binary:



Copy

$ git clone https://github.com/cosmos/gaia.git

$ cd gaia

$ git checkout v7.0.0

$ make install

$ gaiad version

The output of gaiad version should print:



Copy

v7.0.0

Follow along with the gaiad subcommands to query the denom and learn about the channel the tokens came from:



Copy

$ gaiad query ibc-transfer denom-trace 14F9BC3E44B8A9C1BE1FB08980FAB87034C9905EF17CF2F5008FC085218811CC --node https://rpc.cosmos.network:443

Response:



Copy

denom\_trace:

base\_denom: uosmo

path: transfer/channel-141



It happens that an endpoint cannot be reached. If you do not receive an appropriate response, take a look at the [chain registry (opens new window)↗](https://github.com/cosmos/chain-registry/blob/master/cosmoshub/chain.json) and try to use another node for the commands, e.g. to test against the testnet:



Copy

$ gaiad query ibc-transfer denom-trace 14F9BC3E44B8A9C1BE1FB08980FAB87034C9905EF17CF2F5008FC085218811CC --node https://cosmos-testnet-rpc.allthatnode.com:26657

From the terminal output, you now know that there is an IBC port transfer and channel channel-141 that corresponds to the IBC connection between the Hub and Osmosis. To learn the IBC light client behind the port and channel, you need to perform another query.



How do light clients get their name? These are on-chain clients that only keep track of the block hashes of a *counterparty* chain. This allows the chain to create a trustless connection over IBC *without the client duplicating the counterparty chain in full*, making such clients "light" rather than "heavy".

The ibc channel client-state transfer query lists the client details for a specified path:



Copy

$ gaiad query ibc channel client-state transfer channel-141 --node https://rpc.cosmos.network:443

Click the expansion panel to see the detailed response:

****

**Response**

Copy

client\_id: 07-tendermint-259

client\_state:

"@type": /ibc.lightclients.tendermint.v1.ClientState

allow\_update\_after\_expiry: true

allow\_update\_after\_misbehaviour: true

chain\_id: osmosis-1

frozen\_height:

revision\_height: "0"

revision\_number: "0"

latest\_height:

revision\_height: "5901208"

revision\_number: "1"

max\_clock\_drift: 600s

proof\_specs:

- inner\_spec:

child\_order:

- 0

- 1

child\_size: 33

empty\_child: null

hash: SHA256

max\_prefix\_length: 12

min\_prefix\_length: 4

leaf\_spec:

hash: SHA256

length: VAR\_PROTO

prefix: AA==

prehash\_key: NO\_HASH

prehash\_value: SHA256

max\_depth: 0

min\_depth: 0

- inner\_spec:

child\_order:

- 0

- 1

child\_size: 32

empty\_child: null

hash: SHA256

max\_prefix\_length: 1

min\_prefix\_length: 1

leaf\_spec:

hash: SHA256

length: VAR\_PROTO

prefix: AA==

prehash\_key: NO\_HASH

prehash\_value: SHA256

max\_depth: 0

min\_depth: 0

trust\_level:

denominator: "3"

numerator: "1"

trusting\_period: 1206000s

unbonding\_period: 1209600s

upgrade\_path:

- upgrade

- upgradedIBCState



This tutorial only discusses a denom\_trace of a single *hop*. For information on multiple hops and the consequences for frontend services, refer to the [ibc-go docs (opens new window)↗](https://ibc.cosmos.network/main/apps/transfer/overview.html#multiple-hops).

That is a lot of information, but it does not answer the question: *how do you know if this IBC client can be relied upon?*

[#Copy link](https://ida.interchain.io/tutorials/6-ibc-dev/#the-chain-id-and-the-client-id) The chain ID and the client ID

Take a minute to consider this question: *how would you identify a chain?*

An initial response might be the **chain ID**. After all, this is literally the chain identifier. However, the chain ID is **not** a unique identifier. Anybody can start a chain with the same chain ID, so this is not a good parameter by which to verify the identity of the chain you want to connect with.

However, the IBC client ID is generated by the [Cosmos SDK IBC Keeper module (opens new window)↗](https://github.com/cosmos/ibc-go/blob/e012a4af5614f8774bcb595962012455667db2cf/modules/core/02-client/keeper/keeper.go#L56) (ICS-02 does not specify a standard for IBC client IDs). **This means that in the eyes of IBC, a chain is identified by virtue of the client\_id.**

A type of *Chain Name Service* can verify the combination of the chain ID and the client ID. There are a few options that are being used at the moment or in development:

* **Chain Name Service (on-chain, decentralized):**

The [CNS (opens new window)↗](https://github.com/tendermint/cns) aims to be a Cosmos SDK module that the Cosmos Hub will one day run. As a hub through which cross-chain transactions go, it only makes sense for the Cosmos Hub to host critical information on how to reach other chain IDs. CNS is currently under development, with more information to follow.

* **Chain Registry (off-chain, semi-decentralized):**

The [chain registry (opens new window)↗](https://github.com/cosmos/chain-registry) repo is a stopgap solution. Each chain ID has a folder describing its genesis and a list of peers. To claim their chain ID, a blockchain operator must fork the registry repo, create a branch with their chain ID, and submit a pull request to include their chain ID in the official cosmos/registry of chain IDs.

Every chain ID is represented by a folder, and within that folder, a peers.json file contains a list of nodes that you can connect to.

Being able to list all possible blockchain paths is still an unsolved problem. Some ecosystem efforts are already being developed to help bridge this gap. Take for example this [IBC-Cosmos repo by Pulsar (opens new window)↗](https://github.com/PulsarDefi/IBC-Cosmos): it attempts to aggregate all known IBC denoms on all IBC connected chains. They use the following data schema:



Copy

{

"ibc/HASH\_\_CHAIN": {

"chain": String,

"hash": String,

"supply": String,

"path": String,

"origin": {

"denom": String,

"chain": String | List[String] | null

// null if we cannot find this denom on native\_token\_data.json

// list if we could not pick correct chain e.g: [terra, terra2] for uluna

}

}

}

Using this data as a source, one could write an API that allows querying for the path, base denom without querying a node.

A relayer can create a new channel from a newly created blockchain (without an established identity) to another blockchain without revealing too much of its information. Storing path information in the IBC denom that you can trace back and checking the associated client from the channel allows us to estimate of the security guarantees of the asset.

[#Copy link](https://ida.interchain.io/tutorials/6-ibc-dev/#ensure-the-ibc-client-is-not-expired) Ensure the IBC client is not expired

Next to verifying the identity of the chain (or rather of the light client), another thing to consider is whether the light client is expired.

Light clients can expire or become *frozen* if they do not get updated within the TrustingPeriod, or when evidence of misbehavior has been submitted. For example, in the event that the Tendermint consensus fails (if more than 1/3 of validators produce a conflicting block, also known as *double signing*), *and* proof of this consensus failure is submitted on-chain, the IBC client becomes frozen, with a frozen\_height that is nonzero.

In the previous example, the output of gaiad query ibc channel client-state confirms the client status so you know the IBC client is not expired.



To find out how to recover clients that have become expired through submitting a governance proposal, check out the [ibc-go docs (opens new window)↗](https://ibc.cosmos.network/main/ibc/proposals.html).

The latest\_height.revision\_height is the block height when the IBC client was last updated. In the previous example, to ensure that the block height is still up to date, you would have to query the blockchain itself for the block height 5901208 (or the latest block height when you perform the query), and ensure that the timestamp of that block + the trusting\_period of 1209600s/336h/14d is after the current time.

For example, you can verify the IBC client status using the query:



Copy

$ gaiad query block 5901208 --node https://rpc.cosmos.network:443

[#Copy link](https://ida.interchain.io/tutorials/6-ibc-dev/#%F0%9F%8E%89congratulations-%F0%9F%8E%89) 🎉Congratulations!🎉

You have made it all the way through this tutorial and will now be unfazed when encountering an IBC denom in the wild!

synopsis

To summarize, this section has explored:

* The basics of ICS-20 token transfer over IBC.
* How you might encounter the IBC denom notation ibc/... containing a hash of the path information when you interact with assets that were transferred over IBC.
* How to derive an IBC denom, or perform a *denomtrace* query to retrieve the path information and base denom of the asset.
* How to query from the path information to identify chain IDs associated with a light client, making use of the chain registry (and soon CNS).
* How to reason about the security of an IBC asset based on the path information contained in the IBC denom.



If your interest in IBC has been piqued, go to the IBC introduction and learn the intricacies of the IBC protocol and IBC applications, start [here](https://ida.interchain.io/academy/3-ibc/).

previous

[](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/10-wager-denom.html)

**[Play With Cross-Chain Tokens](https://ida.interchain.io/hands-on-exercise/2-ignite-cli-adv/10-wager-denom.html)**

up next

**[Go Relayer](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/1-go-relayer.html)**

[[](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/1-go-relayer.html)](https://ida.interchain.io/hands-on-exercise/5-ibc-adv/1-go-relayer.html)

Rate this Page

icon smile

icon meh

icon frown

Would you like to add a message?

Submit

Thank you for your Feedback!

[](https://ida.interchain.io/ida-course/discord-info.html)

On this page

[ICS-20 - token transfer](https://ida.interchain.io/tutorials/6-ibc-dev/#ics-20-token-transfer)

[How are IBC denoms derived?](https://ida.interchain.io/tutorials/6-ibc-dev/#how-are-ibc-denoms-derived)

[Practical example: denom\_trace](https://ida.interchain.io/tutorials/6-ibc-dev/#practical-example-denom-trace)

[The chain ID and the client ID](https://ida.interchain.io/tutorials/6-ibc-dev/#the-chain-id-and-the-client-id)

[Ensure the IBC client is not expired](https://ida.interchain.io/tutorials/6-ibc-dev/#ensure-the-ibc-client-is-not-expired)

[🎉Congratulations!🎉](https://ida.interchain.io/tutorials/6-ibc-dev/#%F0%9F%8E%89congratulations-%F0%9F%8E%89)

#### **Get Cosmos updates**

Unsubscribe at any time. [Privacy Policy↗](https://v1.cosmos.network/privacy)

     Next

Documentation

[Cosmos SDK](https://docs.cosmos.network/)[Cosmos Hub](https://hub.cosmos.network/)[CometBFT](https://docs.cometbft.com/)[IBC Protocol](https://ibc.cosmos.network/)

Community

[Interchain blog](https://blog.cosmos.network/)[Forum](https://forum.cosmos.network/)[Discord](https://discord.gg/cosmosnetwork)

Contributing

[Source code on GitHub](https://github.com/cosmos/sdk-tutorials)

[](https://ida.interchain.io/)

[Interchain Developer Academy](https://ida.interchain.io/)

**[](https://blog.cosmos.network/)[](https://twitter.com/cosmos)[](https://discord.gg/cosmosnetwork)[](https://www.linkedin.com/company/interchain-foundation/about/)[](https://reddit.com/r/cosmosnetwork)[](https://t.me/cosmosproject)[](https://www.youtube.com/c/CosmosProject)**



Dark mode

† This website is maintained by the Interchain Foundation (ICF). The contents and opinions of this website are those of the ICF. The ICF provides links to cryptocurrency exchanges as a service to the public. The ICF does not warrant that the information provided by these websites is correct, complete, and up-to-date. The ICF is not responsible for their content and expressly rejects any liability for damages of any kind resulting from the use, reference to, or reliance on any information contained within these websites.

Cosmos is a registered trademark of the [Interchain Foundation.](https://interchain.io/)[Privacy](https://v1.cosmos.network/privacy)