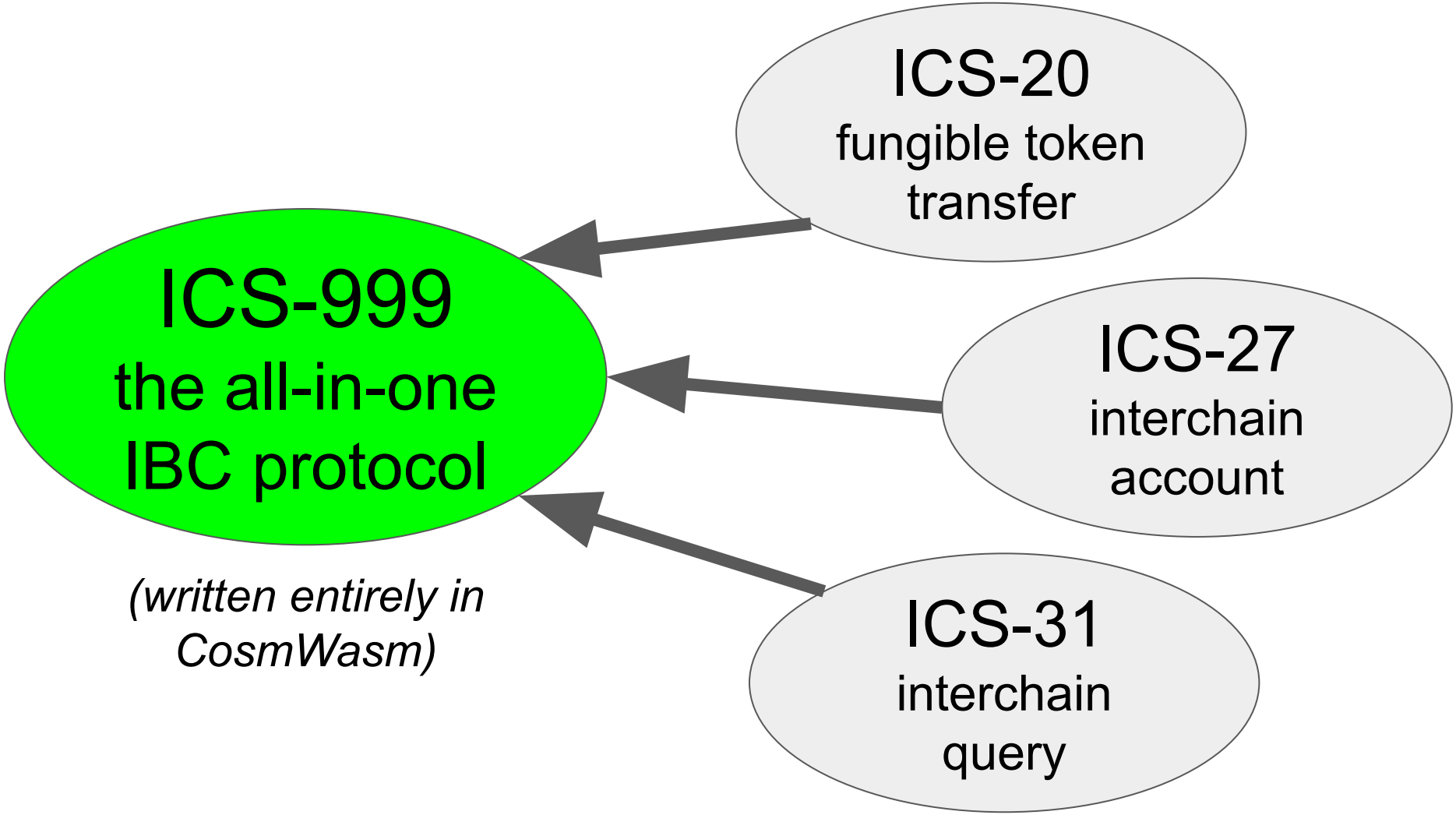
@larry0x

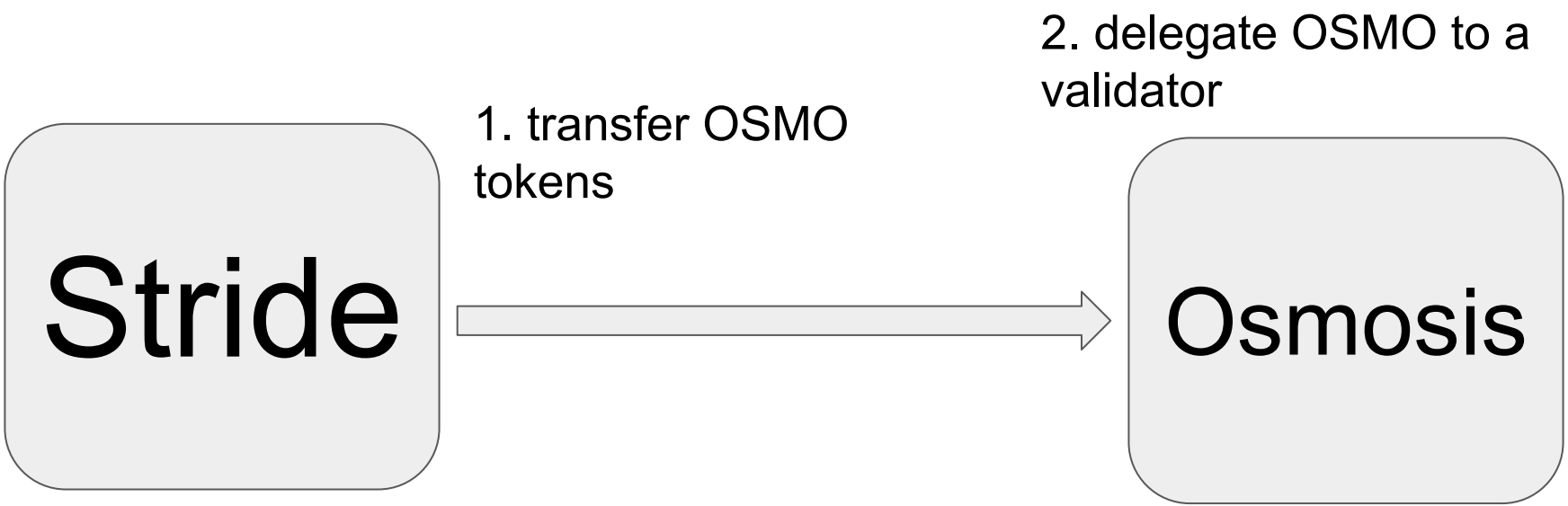
developer at Delphi Labs
contributor to Mars protocol

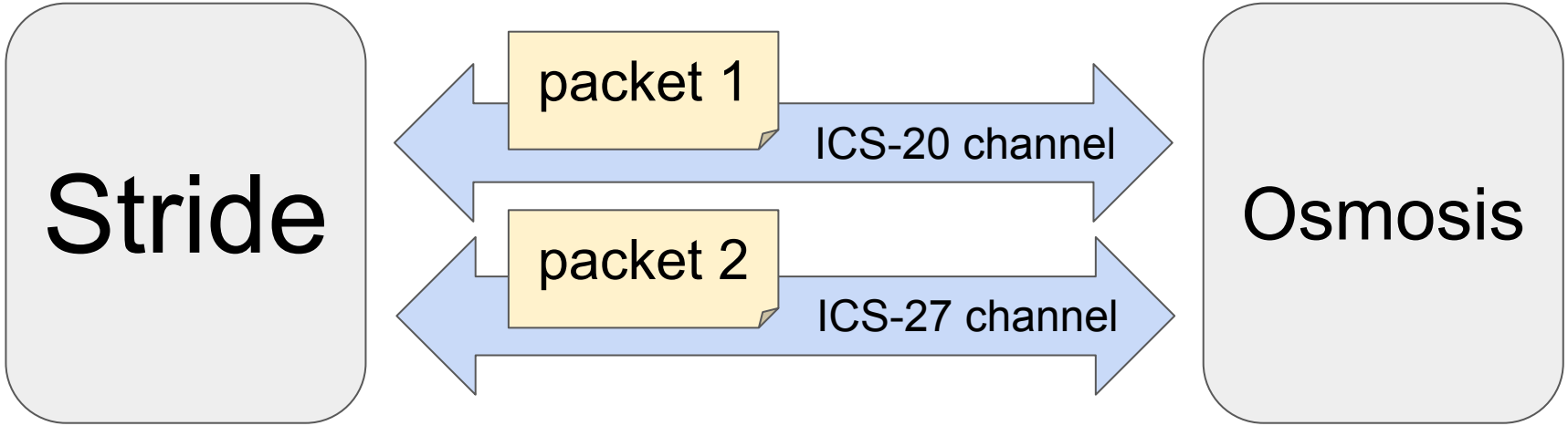
My side projects

- **cw-sdk (name subject to change)**
  - a blockchain framework built on top of CosmWasm
  - create your appchain or app-rollup entirely in CosmWasm
- **ICS-999**
  - the topic of this presentation
- **x/abstractaccount**
  - an account abstraction framework for CosmWasm-enabled chains
  - stay tuned to **my presentation at OsmoCon, July 21!**

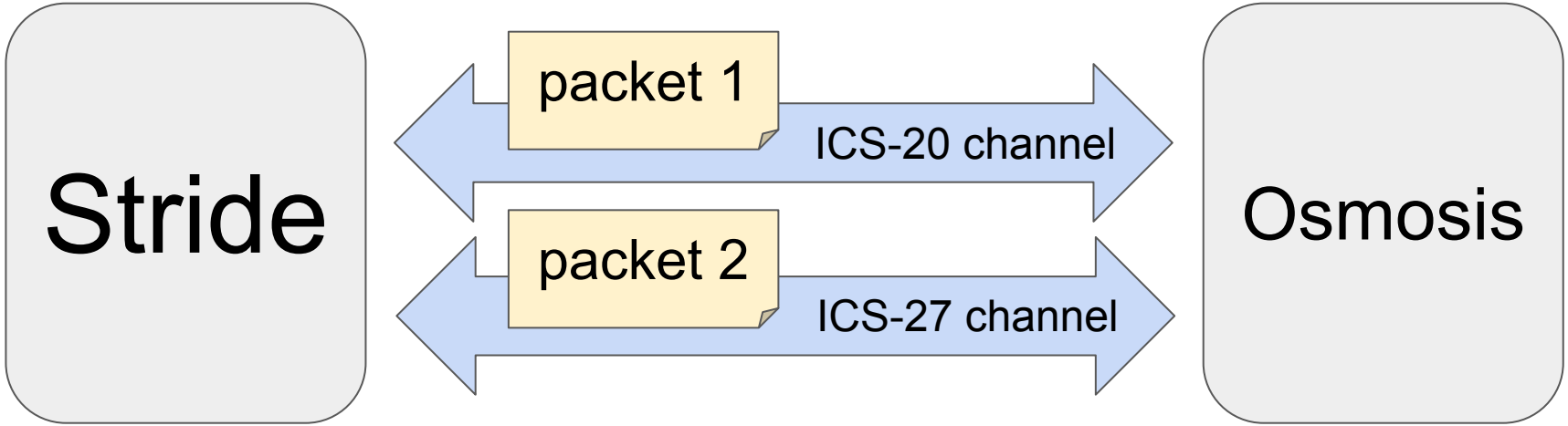

ICS-20
fungible token transfer
ICS-999
the all-in-one IBC protocol
ICS-27
interchain account
(written entirely in CosmWasm)
ICS-31
interchain query

why existing ICS protocols are not very good

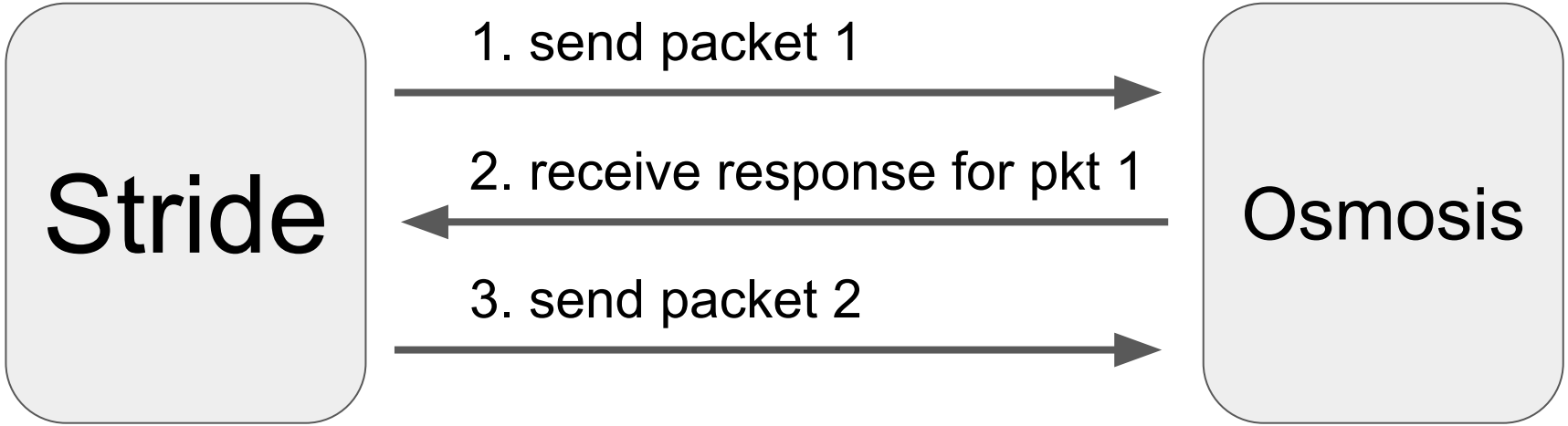

2. delegate OSMO to a validator
1. transfer OSMO tokens
Stride
Osmosis



1. has to be done over 2 packets



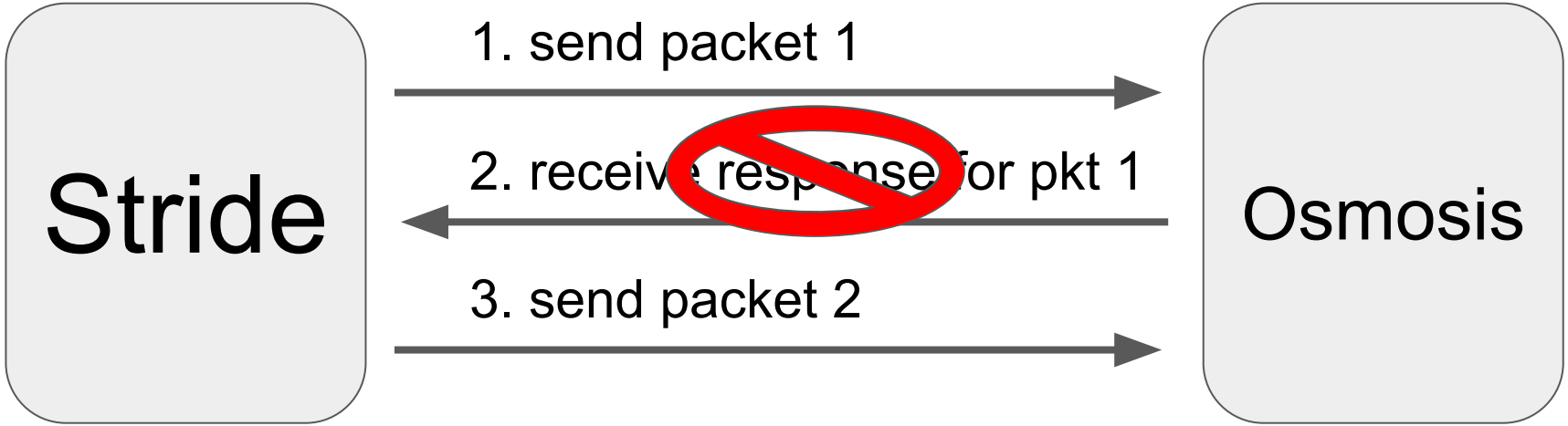
2. there's no ordering between two channels



we have to rely on **async callback pattern** to enforce ordering:

```
sendPacket1().then(response => {
  if response.isTimeout() {
    throw new Error("packet timed out");
  }

  if response.isError() {
    throw new Error("transfer failed");
  }

  return sendPacket2();
});
```

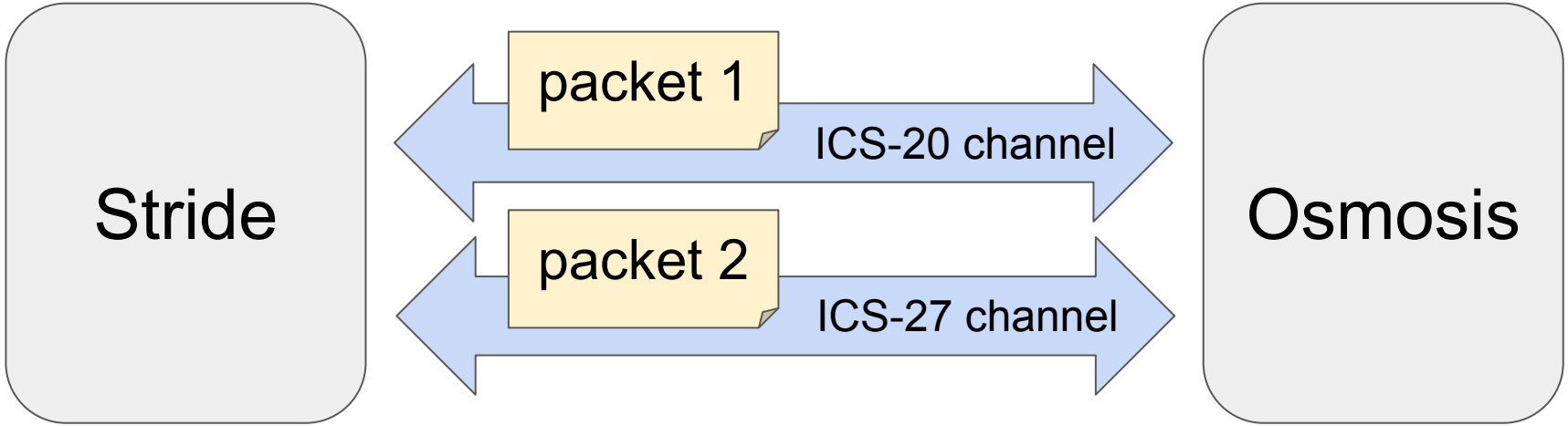
(this doesn't even include the registering ICA step)



3. ICS-20 doesn't provide callback to the sender

possible options:

- Neutron's fork
- Osmosis' ibchooks



4. no **atomicity** between packets

Name	Amount	Total Value
STRD Stride Staking Coin	12,193.473806	$14,144.42 $1.16
stLUNA Stride Staked Luna	7.972705	$5.29 $0.66
stATOM Stride Staked Atom	0.466048	$5.04 $10.82
stOSMO Stride Staked Osmo	0.722690	$0.40 $0.55
stJUNO Stride Staked Juno	0.479023	$0.16 $0.33
stSTARS Stride Staked Stars	4.896692	$0.07 $0.01
stUMEE Stride Staked Umee	61.428090	$0.00 $0.00
stINJ Stride Staked Injective	0.898333	$0.00 $0.00
stEVMOS Stride Staked Evmos	58.317945	$0.00 $0.00
stCMDX Stride Staked CMDX	3.309107	$0.00 $0.00

5. only 1 token transfer per packet

COSMOS
cosmoshub-4

Opened
Channel 24 / 53

#	Osmosis	Cosmos	Operating Period	24h Txs	24h Value
1	channel-0	channel-141	743 Days	2,691 -13.81%	$ 1,130,007.80 -41.78%
2	channel-0	channel-440	743 Days	0 --	0.00 --
3	channel-137	channel-271	533 Days	0 --	0.00 --
4	channel-138	channel-274	533 Days	0 --	0.00 --
5	channel-139	channel-275	488 Days	0 --	0.00 --

6. ICS-20 allows multiple channels between the same two chains

user confusion & stuck funds

7. ICS-27 doesn't have CosmWasm bindings

- protobuf - not fun to work with

8. ICS-27 uses ordered channels

- very fragile
- a huge tradeoff that's not worth it for 99% use cases

ICS-999 solves all these problems!

ICS-20/27	ICS-999
❌ two separate packets needed to 1) transfer tokens, and 2) execute actions via ICA	✅ a single packet
❌ impossible to enforce order between channels	✅ actions within the single packet are ordered
❌ not atomic	✅ atomic
❌ does not provides callback	✅ provides callback
❌ only 1 token per packet	✅ send as many tokens as you want in a single packet
❌ multiple channels between the same two chains allowed	✅ only 1 channel allowed, no user confusion
❌ no CW bindings, protobuf difficult to work with	✅ built entirely in CW, packet data in json
❌ ordered channel, easily drop dead	✅ unordered channel that can never be closed

ICS-999 packet structure (WARNING: subject to change!)

```
struct PacketData {
    sender:  String,
    actions: Vec<Action>,
    traces:  Vec<Trace>,
}

enum Action {
    RegisterAccount {
        salt: Option<Binary>,
    },
    Transfer {
        denom:     String,
        amount:    Uint128,
        recipient: Option<String>,
    },
    Execute(CosmosMsg),
    Query(QueryRequest),
}
```

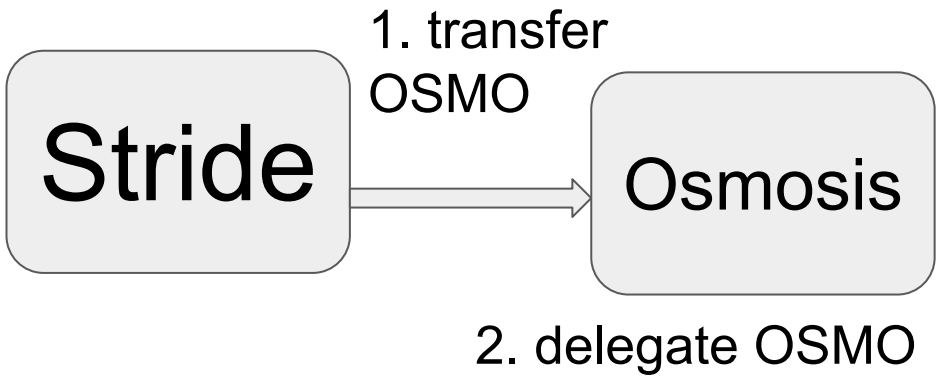
the account who sends the packet

one or more actions that will be executed in order

auxiliary data used in token transfers; auto-populated by the contract

```
enum ExecuteMsg {
    Act {
        connection_id: String,
        actions:       Vec<Action>,
    },
}
```

If Stride is reimplemented in CW…




```
{
  "act": {
    "connection_id": "connection-...",
    "actions": [
      {
        "register_account": {}
      },
      {
        "transfer": {
          "denom": "...",
          "amount": "12345"
        }
      },
      {
        "execute": {
          "staking": {
            "validator": "osmovaloper1...",
            "amount": {
              "denom": "uosmo",
              "amount": "12345",
            }
          }
        }
      }
    ]
  }
}
```

ICS-999 ack structure (WARNING: subject to change!)

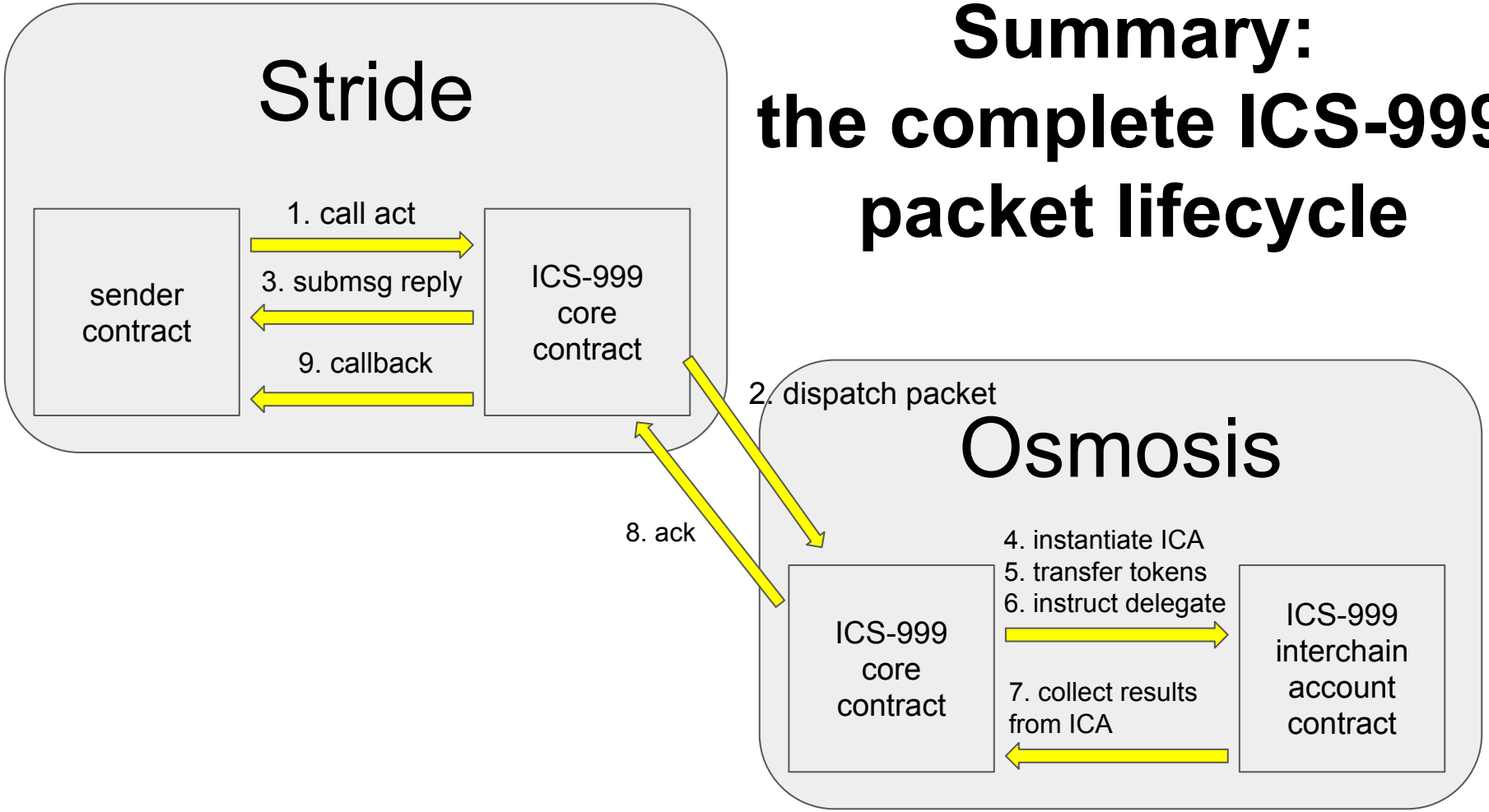
```
struct PacketAck {
    Ok(Vec<ActionResult>),
    Err(String),
}

enum ActionResult {
    RegisterAccount {
        address: String,
    },
    Transfer {
        denom: String,
        // ...
    },
    Execute {
        data: Option<Binary>,
    },
    Query {
        response: Binary,
    },
}
```

if ALL actions succeed, the results are returned. there is one result for each action

if ANY action fails, the entire packet execution fails and is reverted. the sender is provided with the error message

```
enum SenderExecuteMsg {
    PacketAck {
        sequence: u64,
        ack:      PacketAck,
    },
    PacketTimeout {
        sequence: u64,
    },
}
```

Summary: the complete ICS-999 packet lifecycle



Drawbacks and limitations

Drawback 1. token denom isn't compatible with ICS-20

- you need to pick one protocol as "canonical" for your token: either ICS-20 or 999, not both
- existing tokens need to be migrated

Drawback 2. token creation fee

```
struct PacketData {
    sender:              String,
    actions:             Vec<Action>,
    traces:              Vec<Trace>,
    denom_creation_fee: Option<Coin>,
}
```

- many chains charge non-zero fees for token creation!
- the first time a token is transferred, the sender needs to pay the fee

Drawback 2. token creation fee

#489 Set token creation fee to zero; introduce an alternative spam-prevention mechanism EXPEDITED PASSED

Voting Time : 2023-04-21 13:15:35 - 2023-04-22 13:15:35

Final Status : Passed

Yes	No	NoWithVeto	Abstain
89.35%	1.65%	0.01%	8.99%
56,707,023.881027 OSMO	1,048,073.014730 OSMO	4,580.766746 OSMO	5,708,070.257376 OSMO

- Osmosis will have free denom creation in v16
- stargaze, neutron, juno, terra2, (sei, injective, kujira)...???

Drawback 3. no multihop support

```
struct PacketData {
    sender:              String,
    actions:             Vec<Action>,
    traces:              Vec<Trace>,
    denom_creation_fee:  Option<Coin>,
    hops:                Vec<IbcEndpoint>,
}
```

- PFM uses some tricks not available for CW

Drawback 3. no multihop support

Give contracts finer control on IBC packet acknowledgement #1721

Open larry0x opened this issue on Apr 26 · 10 comments

larry0x commented on Apr 26 • edited Contributor

I'm attempting to implement something similar to the packet-forward-middleware as a CosmWasm contract. Unfortunately, as a Go module, PFM is able to control how to write acknowledgement for a packet in a way not available to CW contracts.

- good chance it's coming in CW v2!

Drawback 4. error messages are redacted

```
struct PacketAck {
    Ok(Vec<ActionResult>),
    Err(String),
}
```

- currently, error messages is redacted in CW, due to non-determinism
- you will only get "codespace: wasm, code 5: execute wasm contract failed"

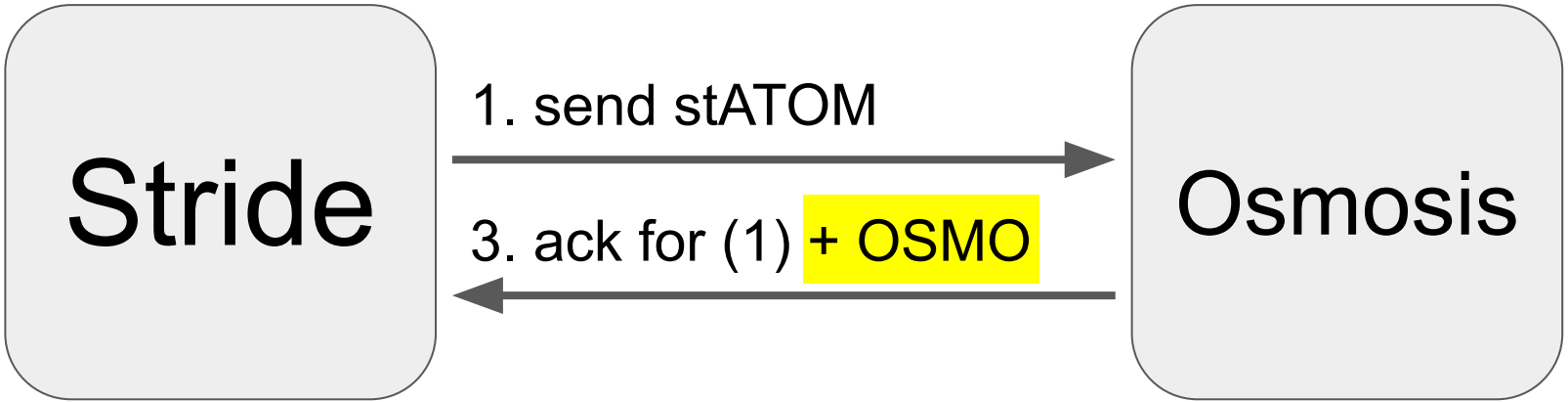
Drawback 5. no relayer fee support

- ICS-29 only supports single-hop, so no point in copying it
- need to come up with a new design that supports multihop!

Ideas for future development

Possible idea: tokens/actions in ack

```
struct PacketAck {
    Ok {
        results: Vec<ActionResult>,
        tokens:  Vec<Coin>,
    },
    Err(String),
}
```



Possible idea: programmable ICA

programmable interchain accounts #3

Open larry0x opened this issue 1 hour ago · 0 comments

larry0x commented 1 hour ago • edited

- WIP: allow users to provide any contract as their ICAs

- If you plan to releasing a token
- If you build wallet apps
- If you build web UIs

LET'S CHAT

```
❯ tokei larry0x/ics999
===============================================================================
 Language            Files        Lines         Code     Comments       Blanks
===============================================================================
 Go                      7         1229          928          121          180
 TOML                    8          214          186            8           20
-------------------------------------------------------------------------------
 Markdown                8          145            0           82           63
 ├─ BASH                 1            4            4            0            0
 ├─ JavaScript           1           58           57            1            0
 ├─ JSON                 1           22           22            0            0
 (Total)                            229           83           83           63
-------------------------------------------------------------------------------
 Rust                   22         2518         2007          134          377
 ├─ Markdown             9          203            3          171           29
 (Total)                           2721         2010          305          406
===============================================================================
 Total                  45         4106         3121          345          640
===============================================================================
```

- ICS-20
  - 11,484 LoC
- ICS-27
  - 15,409 LoC
- neutron/x/interchainqueries
  - 12,742 LoC
- osmosis/x/ibc-hooks
  - 862 LoC

ICS-999: ~2,000 LoC

competitors: 40,497 LoC

CosmWasm = the superior framework for creating IBC application layer protocols