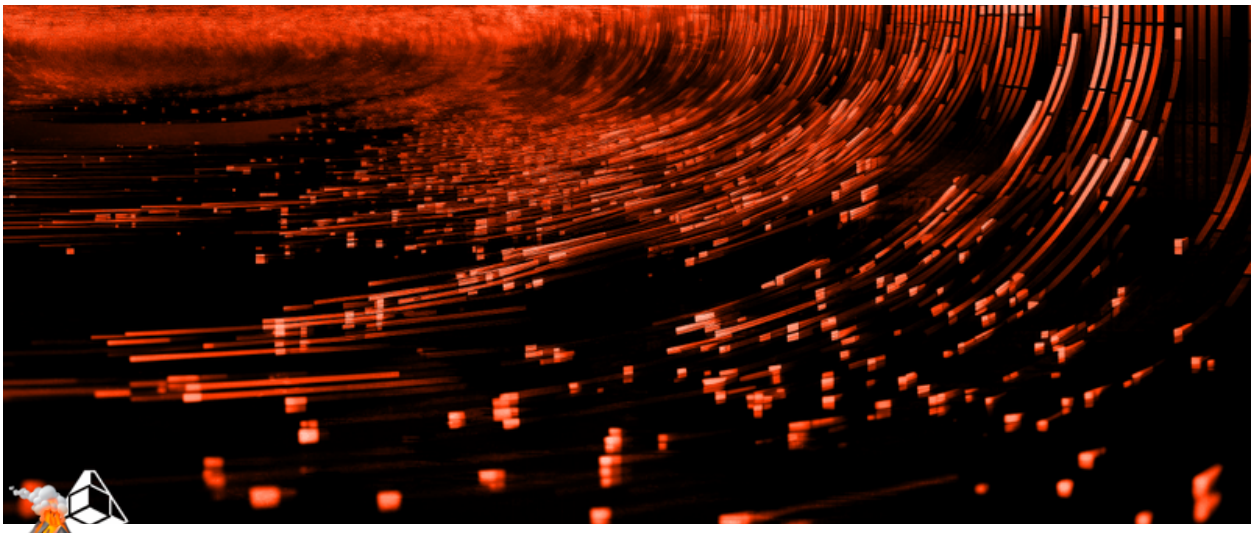

Lava Challenge - gRPC

Step 1 - gRPC Server:

In this challenge, you will build a gRPC server using Golang with the following requirements:

1. The gRPC server should listen for requests from users (you can send the requests using `grpcurl`) and forward the requests to Lava's public RPC endpoint: `lav1.grpc.lava.build:443`.
   - You can test the endpoint using `grpcurl lav1.grpc.lava.build:443 list`.
   - Note: Lava is based on Cosmos SDK.
   - A useful resource (not required for this task): Cosmos documentation.
2. The code should be as generic as possible, allowing the Lava endpoint to be replaced with another gRPC endpoint.
3. The server should listen to `grpcurl` requests.
   - More on grpcurl: grpcurl GitHub repository.
4. Implement the service `cosmos.base.tendermint.v1beta1.Service`.

Ensure that the gRPC requests sent to your server return the same results as the Lava public RPC.

Step 2 - StateTracker:

1. Now that you have a gRPC server running, design a state tracker that will query your server for the latest block information using the `cosmos.base.tendermint.v1beta1.Service.GetLatestBlock` API.
2. After getting the response you should parse the response and extract the following information:

   a. block height - could be found under the key "block" → "height"

   b. block hash - could be found under the key "block_id" → "hash"

2. Parse the information for a duration of 5 blocks, for each block that passes save the data into a JSON file with the following structure:

```
{
 "test_result": [
    {"height": X, "hash" Y}, {"height": X+1, "hash" Z}, etc...
 ]
}
```

Guidelines:

- You may use any modules you like, as long as you explain why.
- All answers should be written in Golang, in a readable fashion, with self-explanatory naming.
- The code should be thoroughly explained with comments.
- Please include a **Readme.md** with your explanations and a walkthrough of how to run the server and state tracker.
- The project should be plug-and-play, without any extra configuration.
- Please complete as many steps as you can within the given time slot. If you cannot finish all the steps, that's okay.
- The code has to be uploaded to a public repository in a version-control platform.

Hints:

- `grpcurl` - `describe` and `list` might help
- `grpcurl` is open source
- Tendermint has precompiled protobufs; there is no need to compile them yourself. You can use the open-source package:

  https://github.com/cosmos/cosmos-sdk