Luxor Interview

Intro

Welcome to the Luxor interview repo! This is a casual coding challenge that should take around 2 hours to complete. You'll have a full 24 hours to submit the code, but we definitely don't expect the task to take the entire day.

Instructions

Once you've completed your code, give us access to the repo or create an MR to the upstream. We'll review the code for completeness, testability, documentation and organization.

The Challenge - A Simple Stratum Server

The goal is to build a simple mock Stratum V1 server. The server stores some information about each request to a postgres database.

For more information on the Stratum protocol check out the Wiki. Stratum is the network protocol used for Bitcoin mining and draws heavily from JSONRPC2. Stratum connections must be durable and long-lived. Anytime a miner is disconnected, efficiency is lost.

https://slushpool.com/help/stratum-protocol#developers

https://en.bitcoin.it/wiki/Stratum_mining_protocol

https://www.jsonrpc.org/specification

The server can bind to any port. Requests must be handled concurrently. Only the following subset of methods need to be implemented.

- `mining.authorize` - Client method. Handles an authorization request.
  - For this simple server, always return a `true` response.
  - Keep track of authorization request still ts in database using your own schema and best judgement.
- `mining.subscribe` - Client method. Lets server know that client is ready to receive work
  - Subscription ID 1 && 2 need to be unique values.
  - ExtraNonce1 is a hex representation of an int64 and *must* be unique accross miners.
  - ExtraNonce2 can be a constant (4 is suitable)
  - Keeping track of the subscription values in the database is important using your own schema and best judgement.
- *Extra Credit* - `mining.notify` - Server method. Periodically notifies the miner of new work.
  - Server should generate a randomized job and broadcast to the list of connected clients
  - *note* for the merkle root parameter in this simple server, and empty array is sufficient.

Ground Rules

The goal is to have functional code that can be the basis of a conversation around design trade-offs during the face-to-face interview process. The most important goal is making a functional piece of code you would want to be reviewed by a colleague.

Include all source code, test code, and a README describing how to build and run the solution.

We think of languages as tools and good engineers can adapt to the right tool for the job. As we primarily program in Golang at Luxor (and it happens to be a great language for this type of project), Golang will be preferred but not required.

Hint: OSX and (most) Linux distros include a telnet client that can be used for ad-hoc testing.

Bonus

Implement a simple client for making and validating requests (also useful for testing!)

Helpful commands

We've provided a `docker-compose.yaml` which should help you bootstrap a Postgres environment.

`docker-compose up` will bring the database up. The files under `./etc/postgres` will be applied automatically

`docker-compose down -v` will bring down the database as well as deleting the volume.

The Compose environment will automatically provision a database:

`dbname=luxor`

`username=luxor`

`password=luxor`

`host=localhost`

`port=5432`

Gominer Client

We've included a testing implementation of a gominer client (MacOS) which can be useful in gaining a functional understanding of how a Stratum server works.

You can also utilize one of our production servers to validate parts of your implementation

```
gominer -url stratum+tcp://btc-us.luxor.tech:6000 -user nickh.test
```

Stuck?

If you get stuck, jump into Discord (https://discord.com/invite/5qPRbDS) or send us an email

(recruiting@luxor.tech)

Helpful resources

Bitcoin wiki - Stratum protocol z-nomp - Open Source NodeJS Mining Pool Slush Reference Stratum Python