Spacemesh Coding Exercise

Peer-to-peer connection management

Our P2P system contains several long-lasting connections between our node and other peers. A connection can either be initiated by the local node or by any remote peer and once created it will serve the node for any message transferring with the remote peer. At any given time, there should be only one connection between the node and a peer. Establishing a connection is a resourceful operation and should only be called when needed and should be parallelised whenever possible.

Build a connection cache that stores all the open connections and supports the following API:

```
Connection* getConnection(int32 ipAddress) // Return existing
connection or create one and store it, this caller should be
blocked until the connection is returned. Should be thread safe.

void onNewRemoteConnection(int32 remotePeer, Connection* conn) //
A callback function that is called whenever a remote peer
establishes a connection with the local node. Should be thread
safe.

void shutdown() // Graceful shutdown, all background workers
should be stopped before this method returns.
```

The code should support unit-testing the connection pool without actually creating the underlying system socket (i.e. by mocking `Connection`)

You can assume that the `Connection` class is already implemented and implements the following interface:

```
IConnection {
      void open()
      void close()
}
```

spacemesh