Client:

Main functionality of the client class is to facilitate storage and retrieval of data. This class must be able to convert arbitrary files into raw bytes and also turn these bytes into CHUNKS in which they will be stored in the STORAGE NODES.

Data: data will be stored as bytes
Chunks: Chunks will be created from data and have functions to allow the storage node and the client to decrypt the bytes and convert them into the proper values

Client should also be able to get a list of all storage nodes and be able to send chunks to the required location based on a hash function.

Storage Node:

Main functionality is to hold onto the data and replicate the data to other nodes in the system. The storage nodes will hold onto chucks in a hashmap for fast retrieval and will use the file name appended with the chunk number (that way the key will be unique). When a Storage node want to enter the network it will make a request the the coordinator, depending on the response the storage node will either shut itself down or join the network.

Storage nodes also need to have a list of all other nodes in order to support the zero-hop hash function. When the client ask for a chunk the storage node should either

1) Respond with a chunk
2) Forward request the the proper storage node

Coordinator:

The coordinator must be able to add nodes to the system and also remove nodes. The coordinator will have no access to the data itself but will be responsible for keeping track of the hash ring as well as propagating the information to the Storage nodes.

Heartbeat:

Coordinator will send heartbeat messages to the storage nodes to see if they are alive or not in the intervals of 5 seconds per heartbeat. The message will be encoded using a proto buff and ping the storage servers will a boolean. If no response comes back in x seconds we can assume the node is dead a remove the storage node from the hash space.