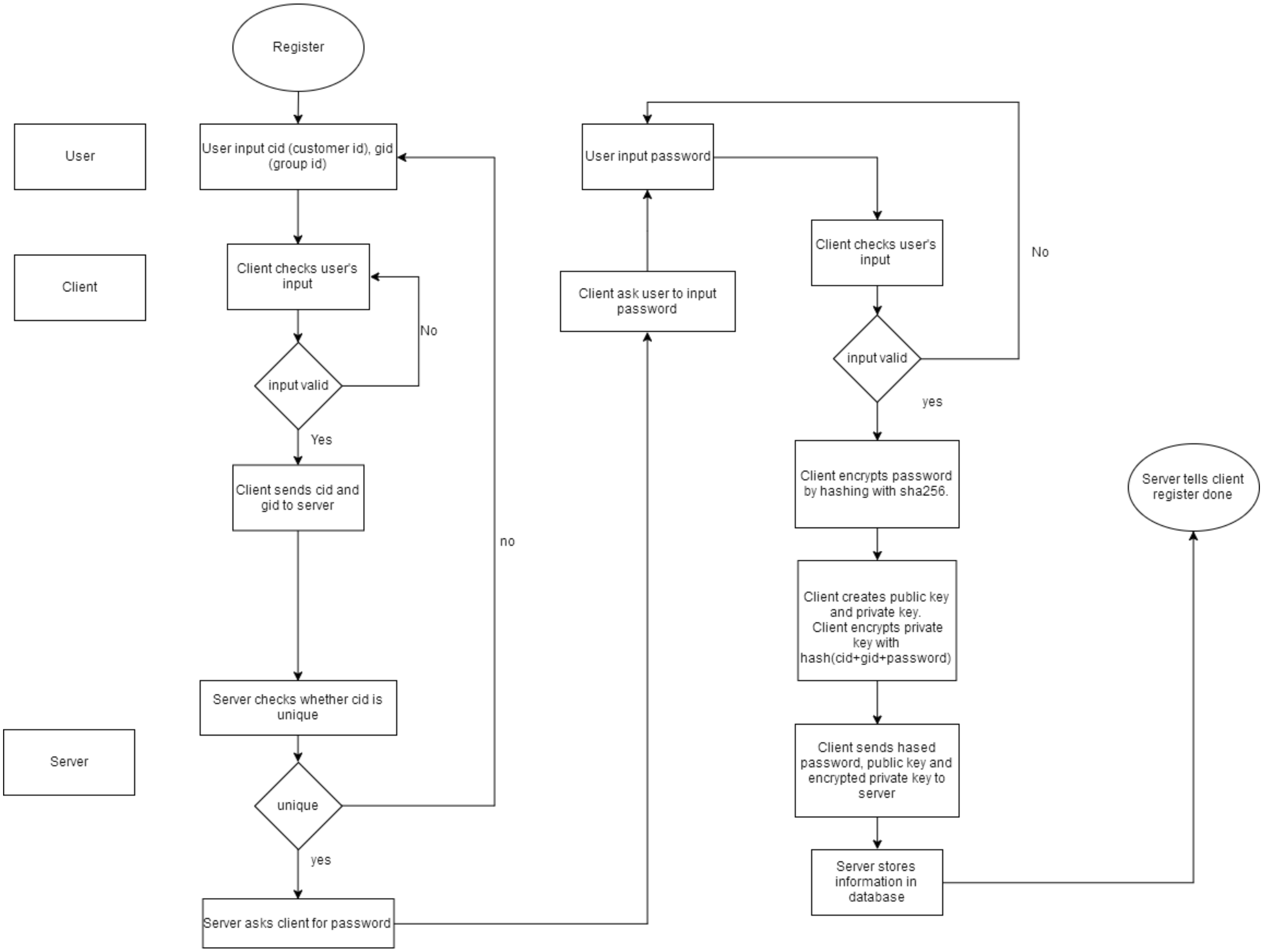
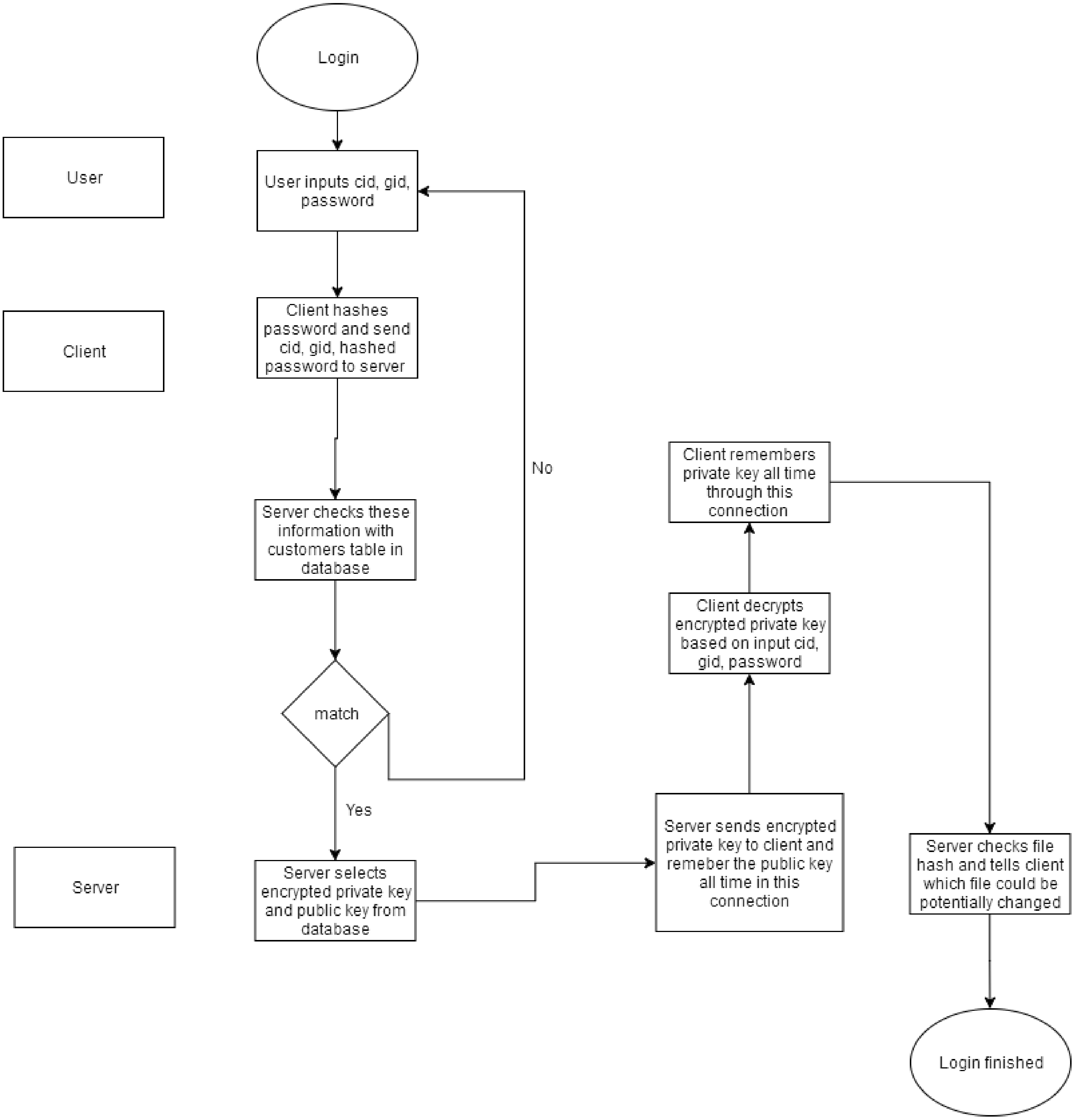
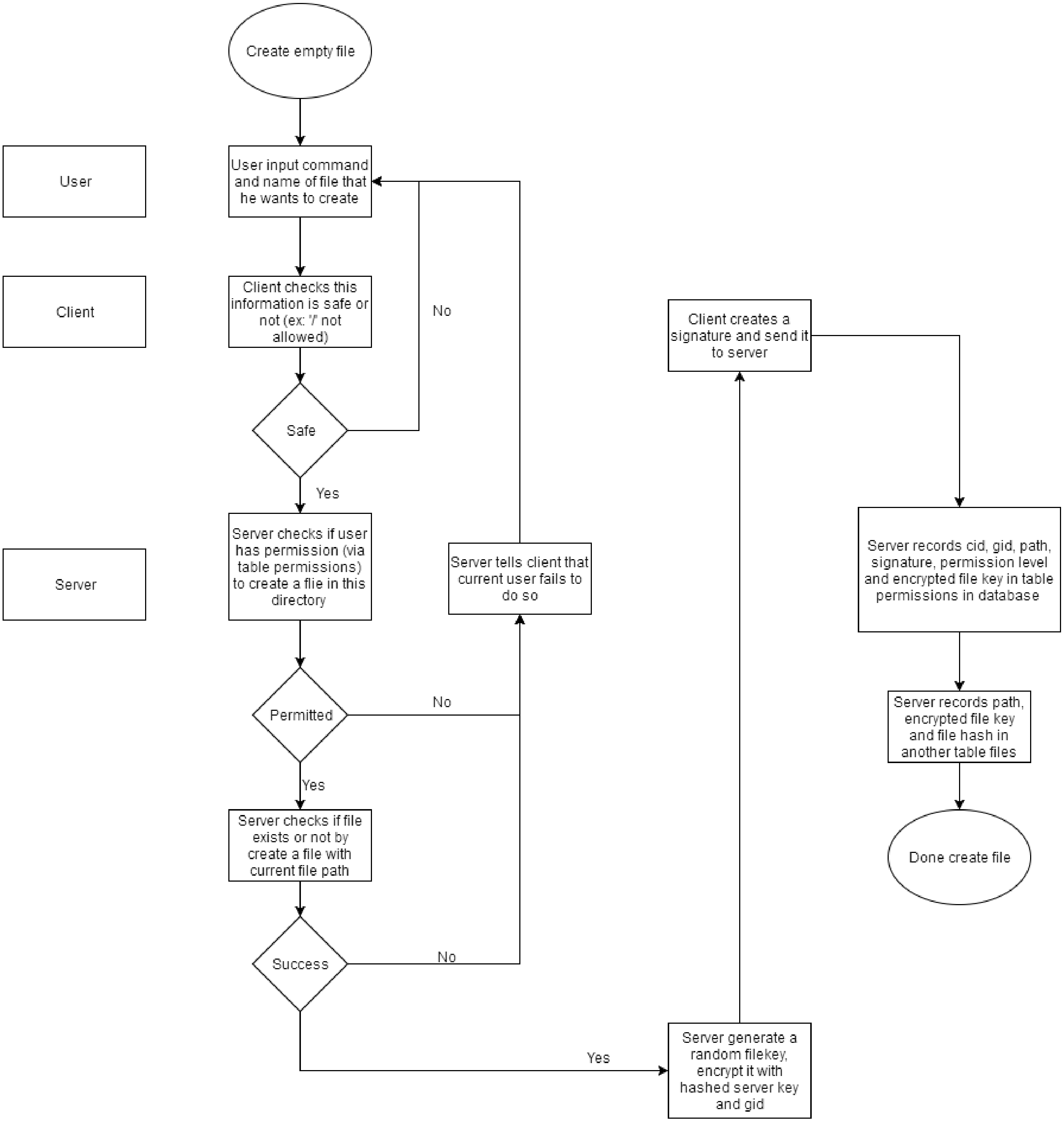
ECE 422 Secure Project Design

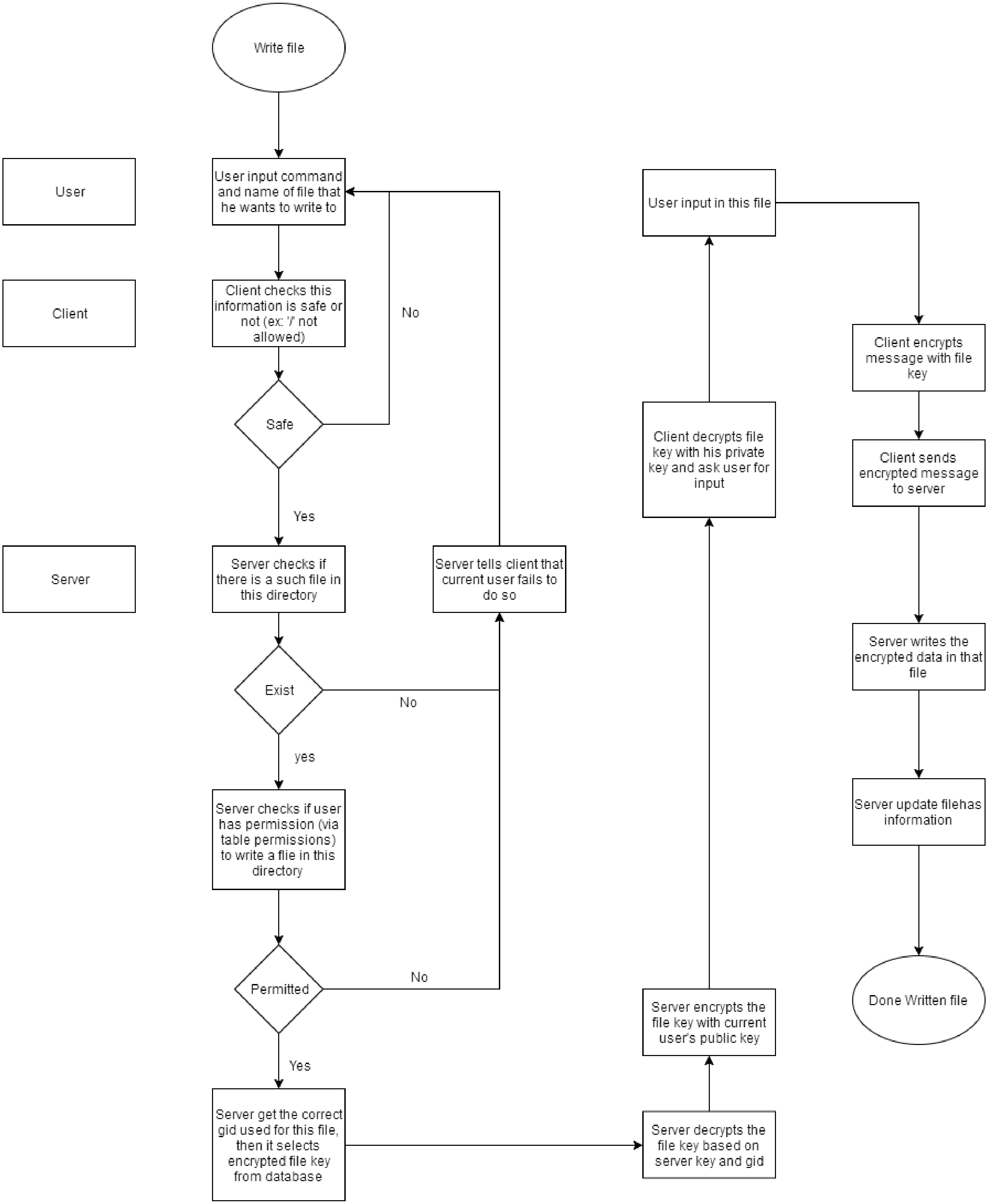
The design of the filesystem operates on a client-server system, involving one trusted server launcher and a variable number of clients. Specifically, security of the system is establshed by involving a server launcher key (which must not be revealed beyond the server launcher) and the client passwords (which is private to each user) for the encryption of files and passwords. Note that for storage and transfer of sensitive informations (such as file encryption keys, passwords) between clients and servers, a combination of hashing and asymmetric encryption are performed to the information as required. To further ensure security, for each file generated, a corresponding unique encryption key is also randomly generated. This minimises the security breach should a specific file key be leaked to a third party. Additionally, to ensure file visibility between permitted groups, and prevent illegal access (out-of-the-system) by third-parties, group keys, generated from server keys, are used to encrypt and decrypt the file encryption keys for storage. The followings are the flow charts showing our design:

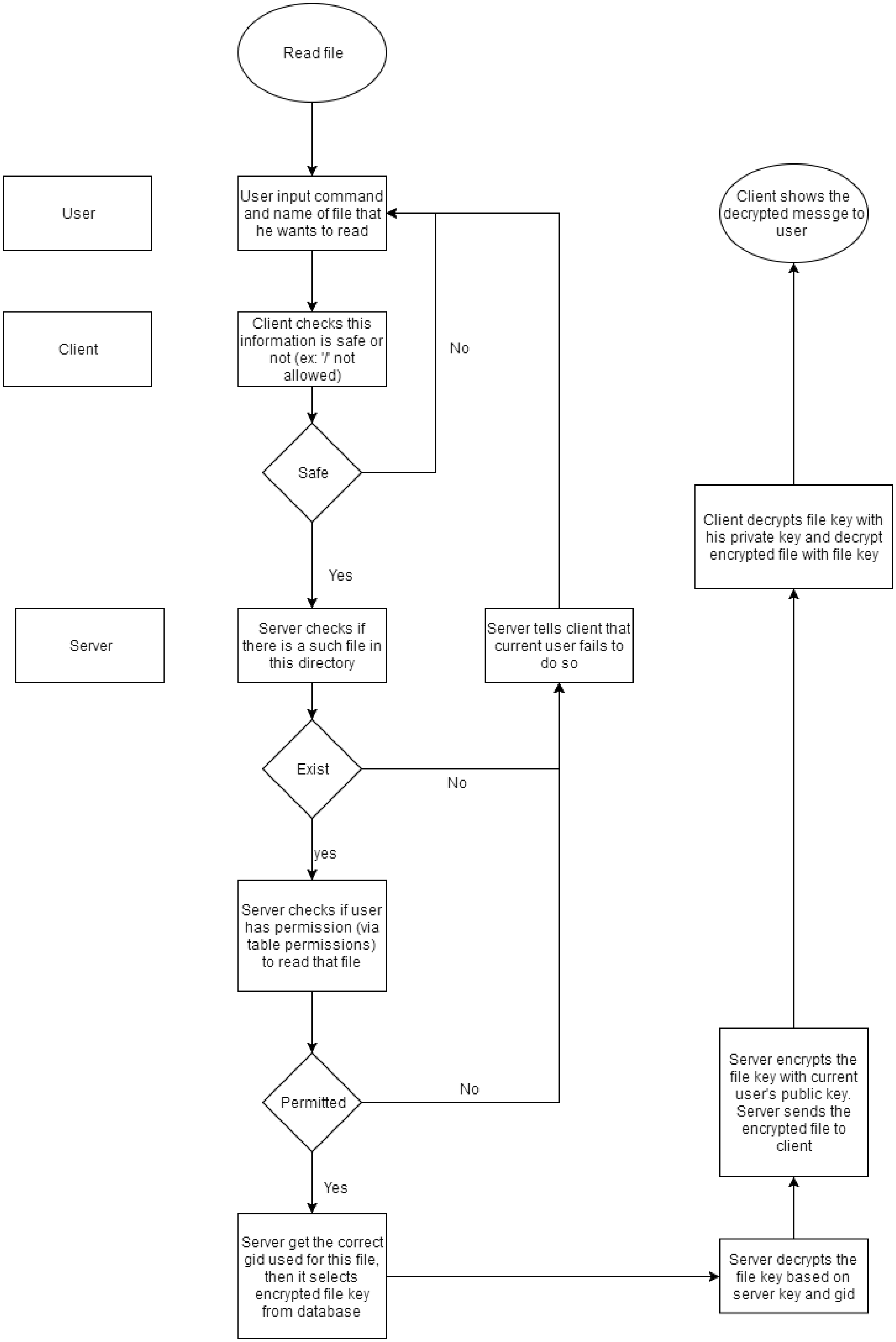


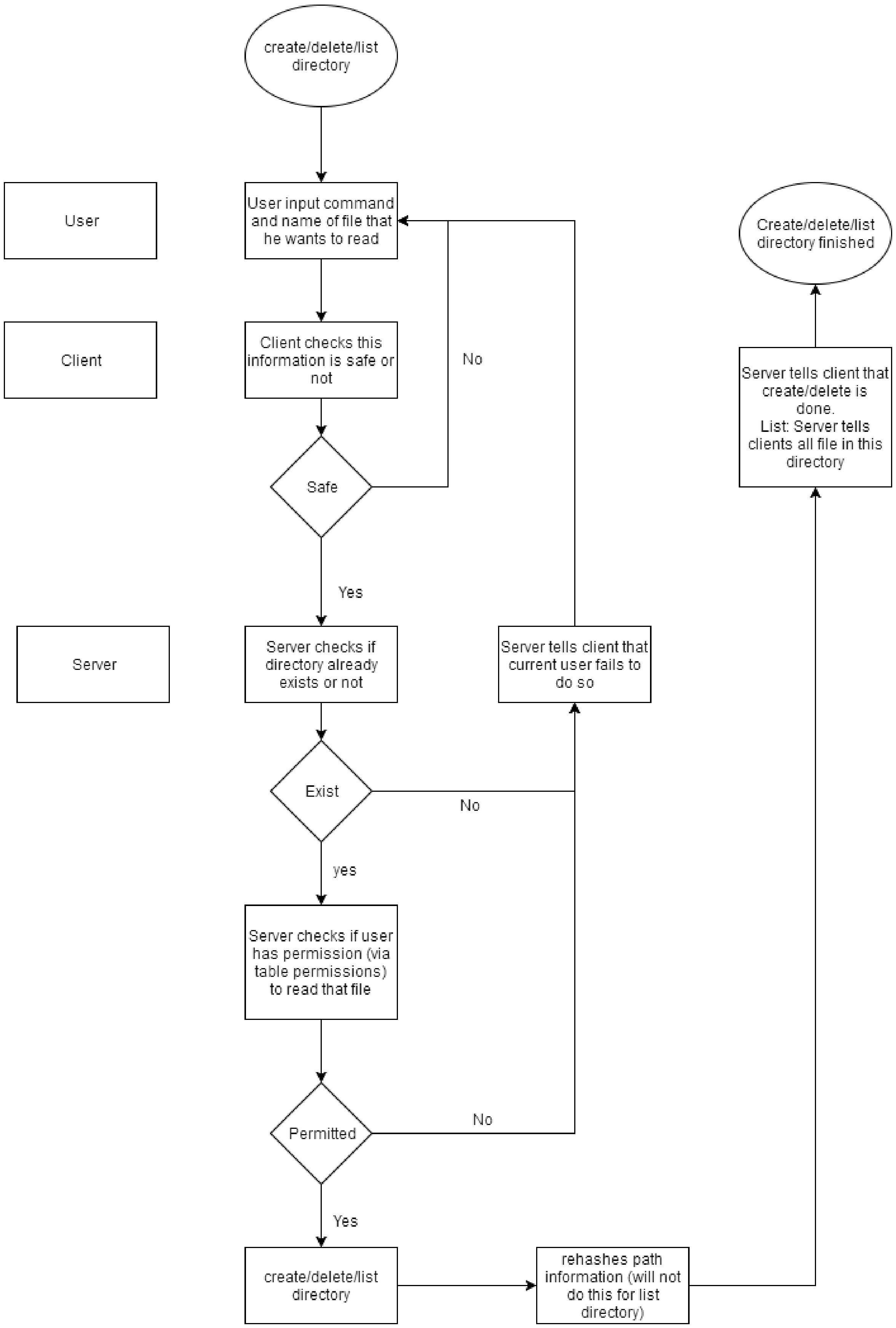
Register
User
User input cid (customer id), gid (group id)
Client
Client checks user's input
input valid
No
Yes
Client sends cid and gid to server
no
Server checks whether cid is unique
Server
unique
yes
Server asks client for password
Client ask user to input password
User input password
Client checks user's input
input valid
No
yes
Client encrypts password by hashing with sha256.
Client creates public key and private key.
Client encrypts private key with hash(cid+gid+password)
Client sends hased password, public key and encrypted private key to server
Server stores information in database
Server tells client register done

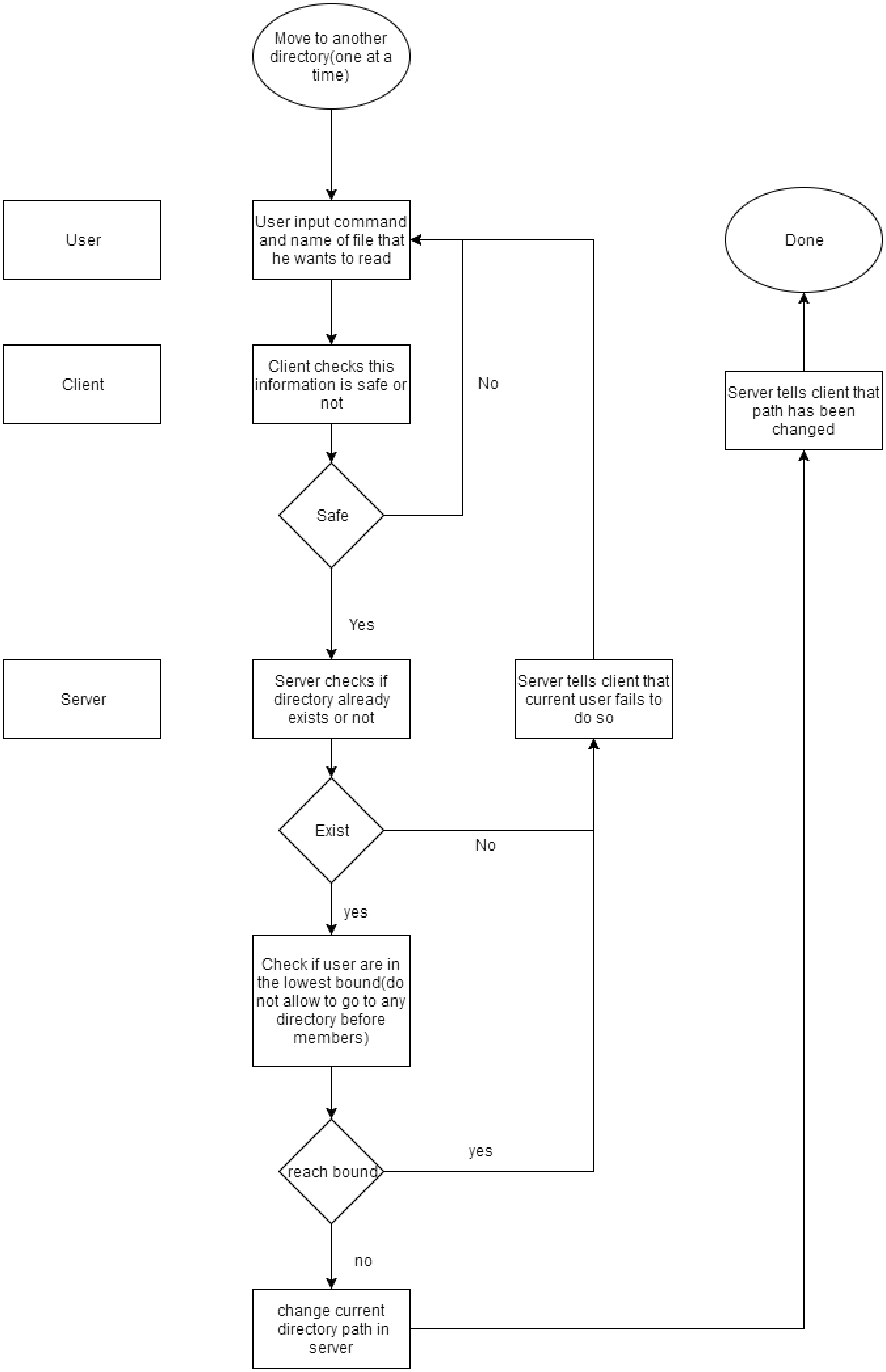

Login
User
User inputs cid, gid, password
Client
Client hashes password and send cid, gid, hashed password to server
No
Server checks these information with customers table in database
match
Yes
Server
Server selects encrypted private key and public key from database
Server sends encrypted private key to client and remeber the public key all time in this connection
Client decrypts encrypted private key based on input cid, gid, password
Client remembers private key all time through this connection
Server checks file hash and tells client which file could be potentially changed
Login finished


Create empty file
User
User input command and name of file that he wants to create
Client
Client checks this information is safe or not (ex: '/' not allowed)
Safe
No
Yes
Server
Server checks if user has permission (via table permissions) to create a flie in this directory
Permitted
No
Yes
Server checks if file exists or not by create a file with current file path
Success
No
Yes
Server tells client that current user fails to do so
Server generate a random filekey, encrypt it with hashed server key and gid
Client creates a signature and send it to server
Server records cid, gid, path, signature, permission level and encrypted file key in table permissions in database
Server records path, encrypted file key and file hash in another table files
Done create file


Write file
User
User input command and name of file that he wants to write to
Client
Client checks this information is safe or not (ex: '/' not allowed)
No
Safe
Yes
Server
Server checks if there is a such file in this directory
Server tells client that current user fails to do so
Exist
No
yes
Server checks if user has permission (via table permissions) to write a flie in this directory
No
Permitted
Yes
Server get the correct gid used for this file, then it selects encrypted file key from database
Server decrypts the file key based on server key and gid
Server encrypts the file key with current user's public key
Client decrypts file key with his private key and ask user for input
User input in this file
Client encrypts message with file key
Client sends encrypted message to server
Server writes the encrypted data in that file
Server update filehas information
Done Written file


Read file
User
User input command and name of file that he wants to read
Client
Client checks this information is safe or not (ex: '/' not allowed)
Safe
No
Yes
Server
Server checks if there is a such file in this directory
Server tells client that current user fails to do so
Exist
No
yes
Server checks if user has permission (via table permissions) to read that file
Permitted
No
Yes
Server get the correct gid used for this file, then it selects encrypted file key from database
Server decrypts the file key based on server key and gid
Server encrypts the file key with current user's public key. Server sends the encrypted file to client
Client decrypts file key with his private key and decrypt encrypted file with file key
Client shows the decrypted messge to user


create/delete/list directory
User
User input command and name of file that he wants to read
Create/delete/list directory finished
Client
Client checks this information is safe or not
No
Server tells client that create/delete is done.
List: Server tells clients all file in this directory
Safe
Yes
Server
Server checks if directory already exists or not
Server tells client that current user fails to do so
Exist
No
yes
Server checks if user has permission (via table permissions) to read that file
Permitted
No
Yes
create/delete/list directory
rehashes path information (will not do this for list directory)


Move to another directory(one at a time)
User
User input command and name of file that he wants to read
Done
Client
Client checks this information is safe or not
No
Server tells client that path has been changed
Safe
Yes
Server
Server checks if directory already exists or not
Server tells client that current user fails to do so
Exist
No
yes
Check if user are in the lowest bound(do not allow to go to any directory before members)
yes
reach bound
no
change current directory path in server