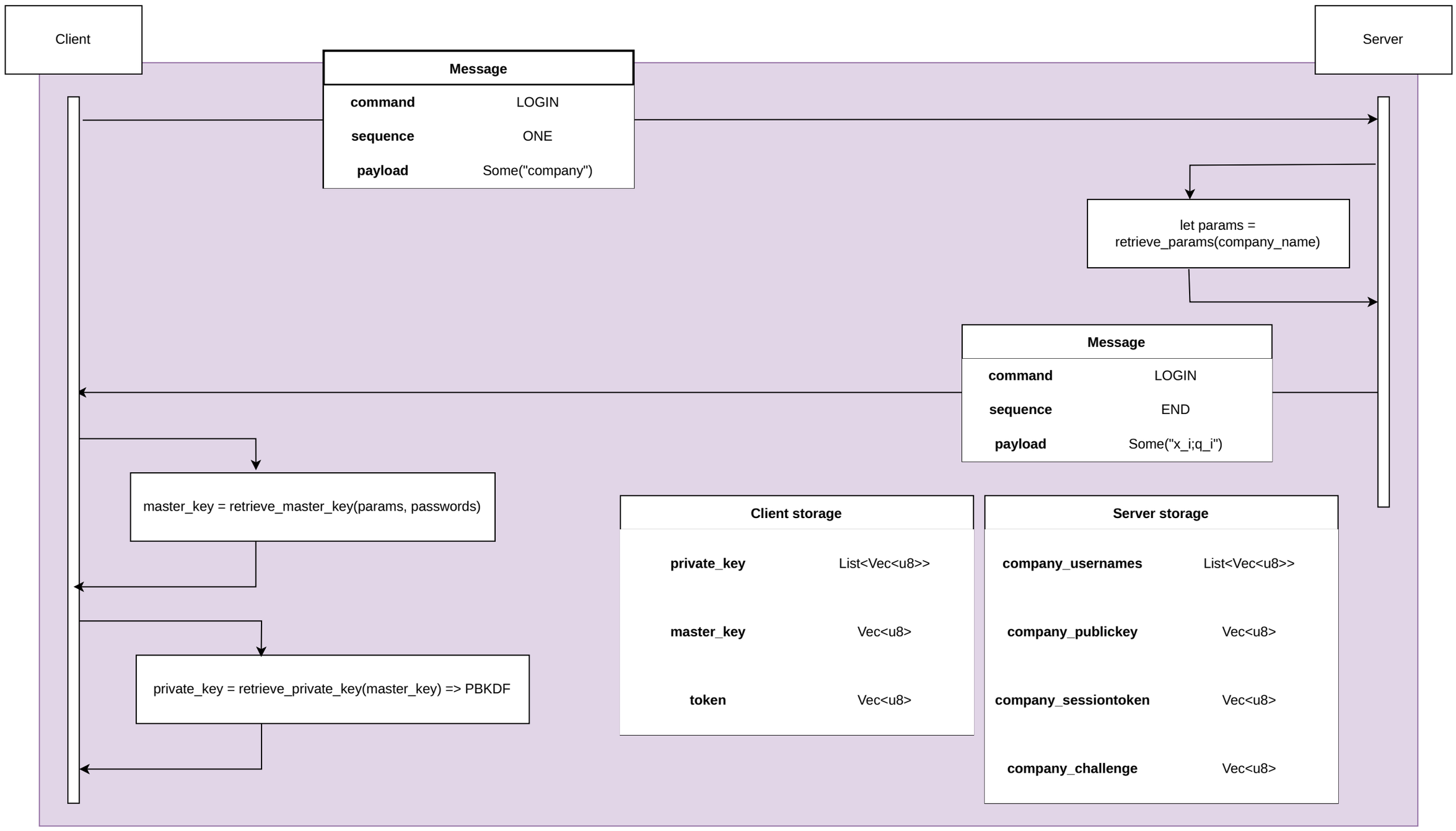
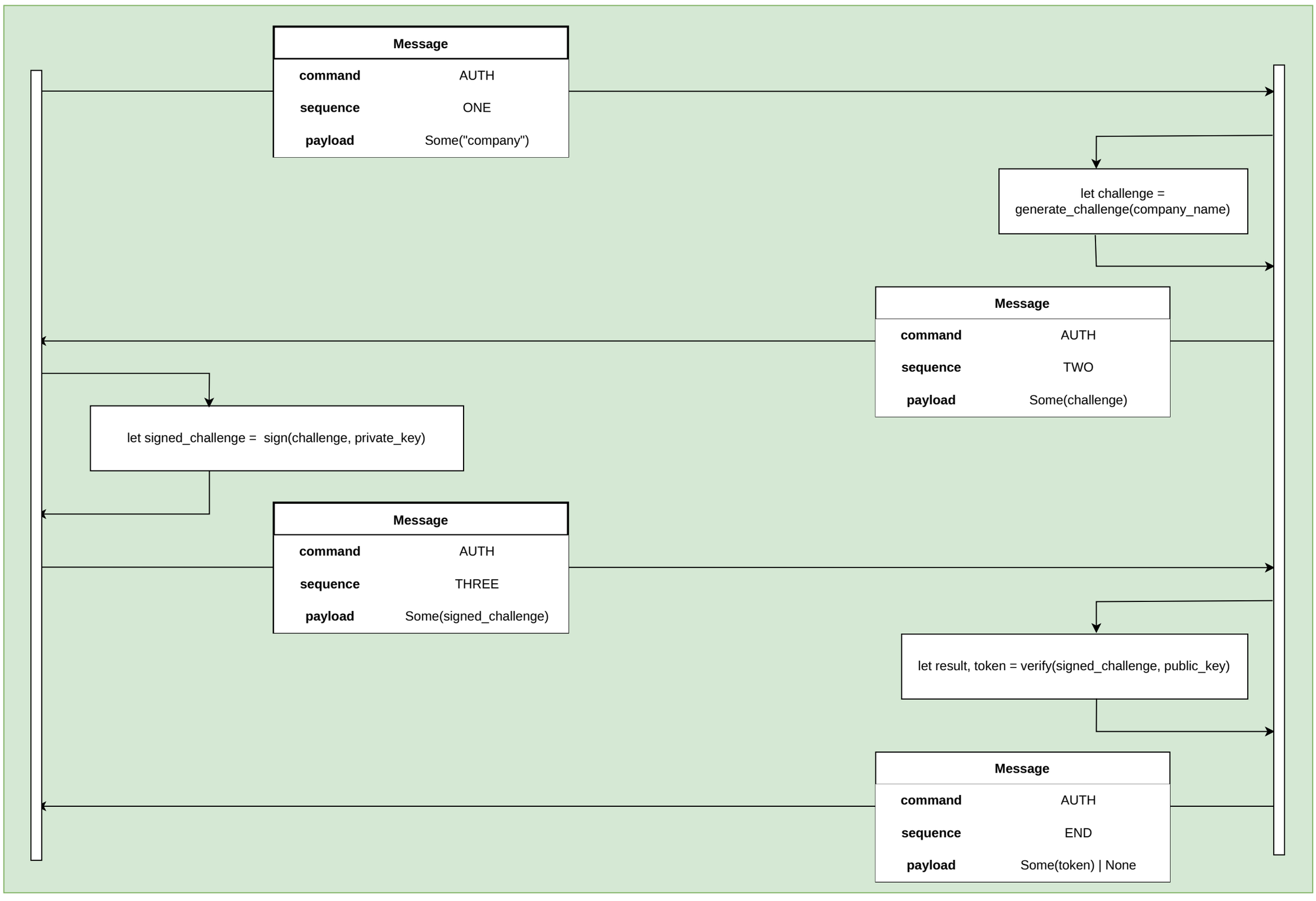
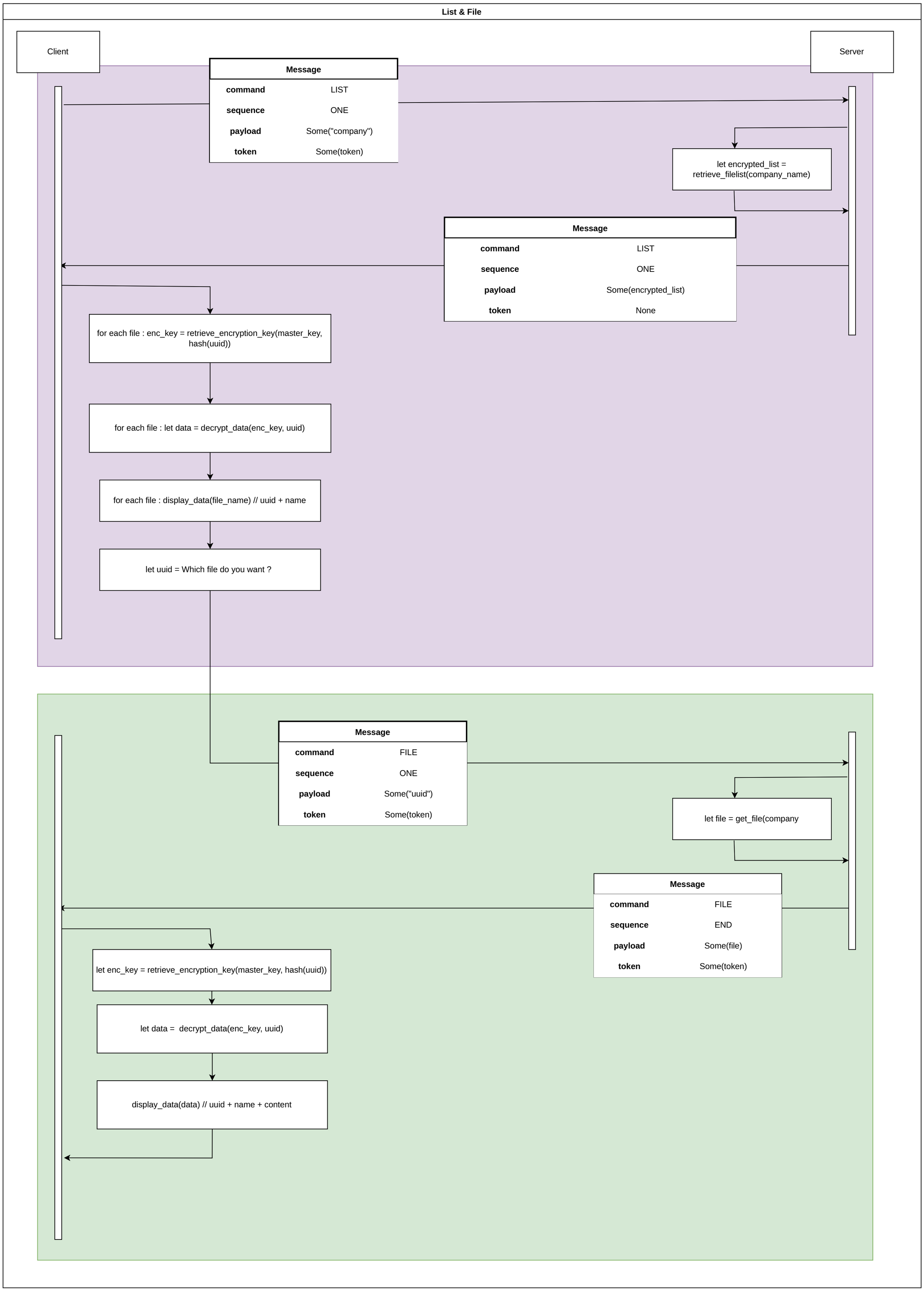
Login & Auth


Client
Server
Message
command LOGIN
sequence ONE
payload Some("company")
let params = retrieve_params(company_name)
Message
command LOGIN
sequence END
payload Some("x_i;q_i")
master_key = retrieve_master_key(params, passwords)
private_key = retrieve_private_key(master_key) => PBKDF
Client storage
private_key List<Vec<u8>>
master_key Vec<u8>
token Vec<u8>
Server storage
company_usernames List<Vec<u8>>
company_publickey Vec<u8>
company_sessiontoken Vec<u8>
company_challenge Vec<u8>


Message
command AUTH
sequence ONE
payload Some("company")
let challenge = generate_challenge(company_name)
Message
command AUTH
sequence TWO
payload Some(challenge)
let signed_challenge = sign(challenge, private_key)
Message
command AUTH
sequence THREE
payload Some(signed_challenge)
let result, token = verify(signed_challenge, public_key)
Message
command AUTH
sequence END
payload Some(token) | None


List & File
Client
Server
Message
command LIST
sequence ONE
payload Some("company")
token Some(token)
let encrypted_list = retrieve_filelist(company_name)
Message
command LIST
sequence ONE
payload Some(encrypted_list)
token None
for each file : enc_key = retrieve_encryption_key(master_key, hash(uuid))
for each file : let data = decrypt_data(enc_key, uuid)
for each file : display_data(file_name) // uuid + name
let uuid = Which file do you want ?
Message
command FILE
sequence ONE
payload Some("uuid")
token Some(token)
let file = get_file(company
Message
command FILE
sequence END
payload Some(file)
token Some(token)
let enc_key = retrieve_encryption_key(master_key, hash(uuid))
let data = decrypt_data(enc_key, uuid)
display_data(data) // uuid + name + content