UNIVERSITÀ DI PISA

**COMPUTER ENGINEERING**

FOUNDATION OF CYBERSECURITY

A.A. 2022-2023

# A SECURE BANK APPLICATION

STUDENTS:

FEDERICO CAVEDONI

GIOVANNI BARBIERI

## CONTENT

# 1 INTRODUCTIONS

Our project consists of a cryptographically secure banking application, developed in C++ for Linux systems.
The application is made up of a Server, which impersonates the bank, and one or more clients that contact the bank via TCP protocol to perform 3 functions made available:

- Check Account ID and Balance
- Perform a Transfer
- Read transfer history.

The project has been divided into subdirectories for readability and convenience.
In communication between client and server, the length of the packet is sent first and then the packet is sent, so as to correctly manage the length of the packets.

## 1.1 SERVER

The 'Server' subdirectory contains the server Main (MainServer.cpp) and the file where all the Server functions are contained (Server.cpp).
The purpose of the server is to listen and serve the clients, providing the functions of the banking application (Read Account Id and Balance, Make a Transfer, Read the history of the transactions).

## 1.2 CLIENT

The 'Client' subdirectory contains the client-related main (MainClient.cpp) and the various client-related functions (Client.cpp).
The client must first authenticate with the server via the Diffie Hellman protocol, then must log in with Username and Password, enter the password for the private key and can thus perform the functions made available to the server.

## 1.3 DTO

The 'DTO' (Data transfer Object) subdirectory is a directory that contains the 'Transaction' and 'User' classes, classes useful in managing the user and the transactions carried out by the client.

## 1.4 FILEMANAGER

In the 'FileManager' subdirectory we find the functions for managing the file.
In particular, functions that allow us to check the password, saved hashed in a specific file, or to read the transaction history or the ID and balance of a user.

## 1.5 PACKETS

The 'Packets' subdirectory contains the 3 types of packages used in the project:

- StartPacket
- AuthenticationPacket
- GenericPacket

The startPacket is used by the Client to initiate the connection with the server.
The authenticationPacket is used by both the client and the server to authenticate each other.
The GenericPacket is used as a generic message between server and client after successful authentication.

## 1.6 UTIL

The 'Util' subdirectory contains two files that are absolutely essential for the project.
The first is UtilFunctions.cpp, which contains various functions used throughout the project.
We then have CryptoFunctions.cpp which contains all the cryptography-related functions used within the project, which allowed us to make the banking application cryptographically secure.

## 1.7 USERS

The 'Users' subdirectory contains 3 types of files for each user:

- username.txt
- password.txt
- transaction_enc.bin

The Username.txt file contains various information related to the username, such as Account Id, username, balance and salt.
The transaction_enc.bin file contains the transactions, which are encrypted for security reasons.
The password.txt file contains the password salted and hashed, also for security reasons.

## 1.8 USERNAME_KEYS

The username_keys subdirectories contain the public and private keys of the client users.
This folder is present only in the corresponding clients.
The server's public and private keys are contained in the 'Server' subdirectory.

## 2 PROTOCOLS

To develop this project and implement the cryptographic functions we used the OpenSSL 3.0 library.
We generated for all users and the server a pair of keys. These keys were created using the openssl library issuing the following commands from the Linux terminal:

- openssl genpkey -algorithm RSA -aes256 -out private_key.pem
- openssl rsa -pubout -in private_key.pem -out public_key.pem

To perform the initial authentication between server and client we used the Diffie Hellman protocol thanks to which two shared secrets were derived, one used as a symmetric key for communications, using the AES256 protocol, and one to generate the HMAC digest using the SHA512 protocol.
The packets exchanged after authentication are therefore encrypted with the key to ensure confidentiality.
A signed HMAC digest is added to the package to ensure integrity and authenticity.
Then we have the counter which guarantees freshness.
User passwords are saved salted and hashed, to avoid saving keys in the clear and adding randomness to the hash.
Transactions are encrypted with a specific key dedicated to the files and decrypted when necessary.

NOTE: The keys sent to establish the shared secret are serialized and deserialized using the BIO structures provided by the openssl library.

# 3 PACKETS FORMAT

We have created 3 different types of packages to handle communication between client and server.

## 3.1 STARTPACKET

The startpacket is used by the client to begin the authentication process on the server. The client generates the two Diffie-Hellmann parameters and sends username length, parameter length, username and parameters in clear.

| | | | | | |
|---|---|---|---|---|---|
| Username len | symmetric key len | hmac key len | username | symmetic key param | hmac key param |

## 3.2 AUTHENTICATIONPACKET

The Authentication packet is used from client to server and from server to client to authenticate each other. The server uses this packet to send its network parameters in the clear, so that the server can also construct symmetric keys.
The signature is also sent, which is formed by the concatenation of the two keys first hashed and then signed, so as to guarantee authentication and integrity.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| iv len | symmetric key len | hmac key len | symm hash len | sign len | hmac hash len | iv | symmetric param | hmac param | sign |

## 3.3 GENERICPACKET

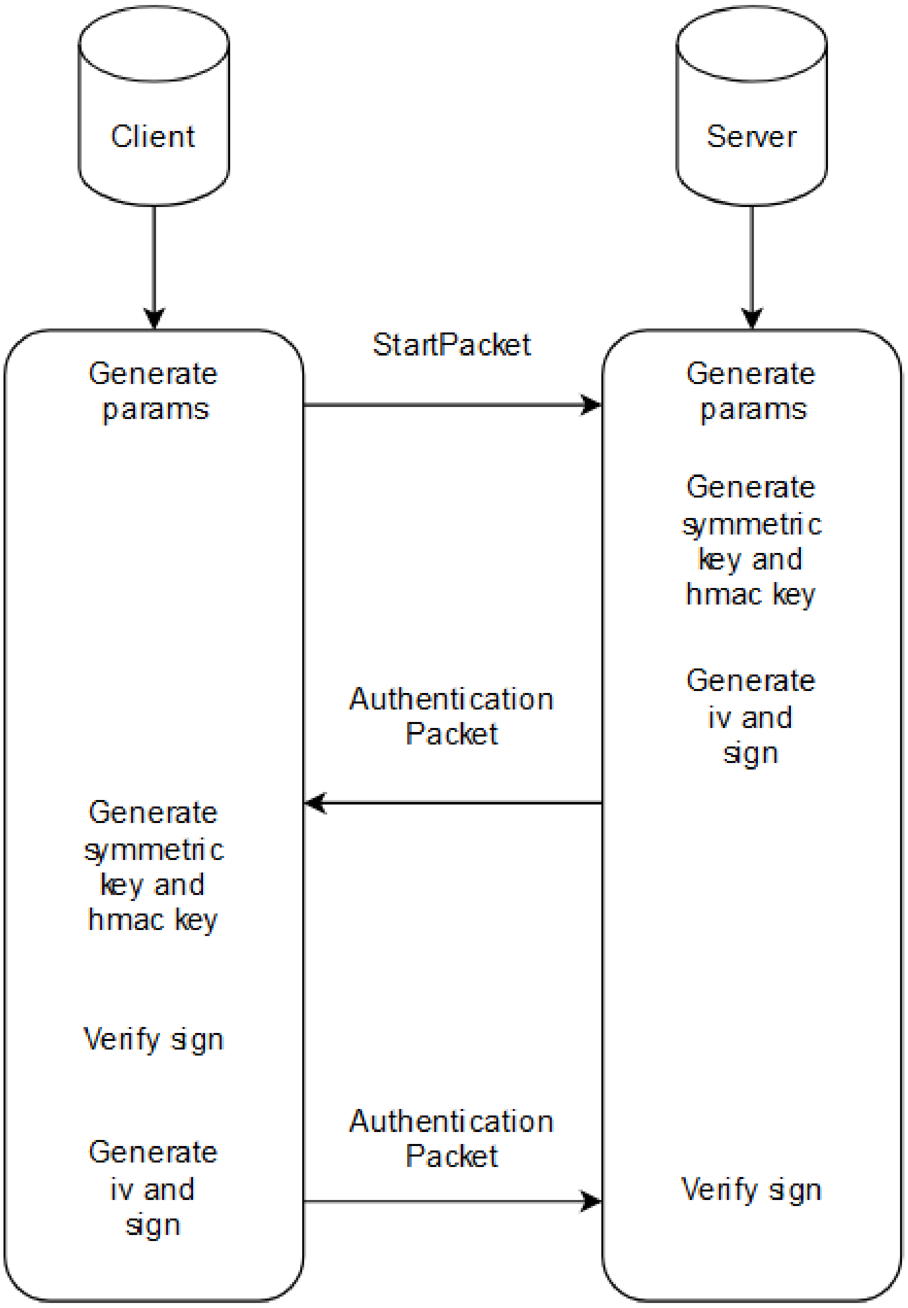
The generic packet is used as a generic communication packet between client and server to manage the various functions offered by the server.
The packet is composed of the iv, used to encrypt and decrypt, the ciphertext encrypted with the symmetric key and the HMAC, used to guarantee integrity, encrypted with the hmac key.

| | | | | | |
|---|---|---|---|---|---|
| iv len | ciphertext len | HMAC len | iv | ciphertext | HMAC |

# 4 WORKFLOW

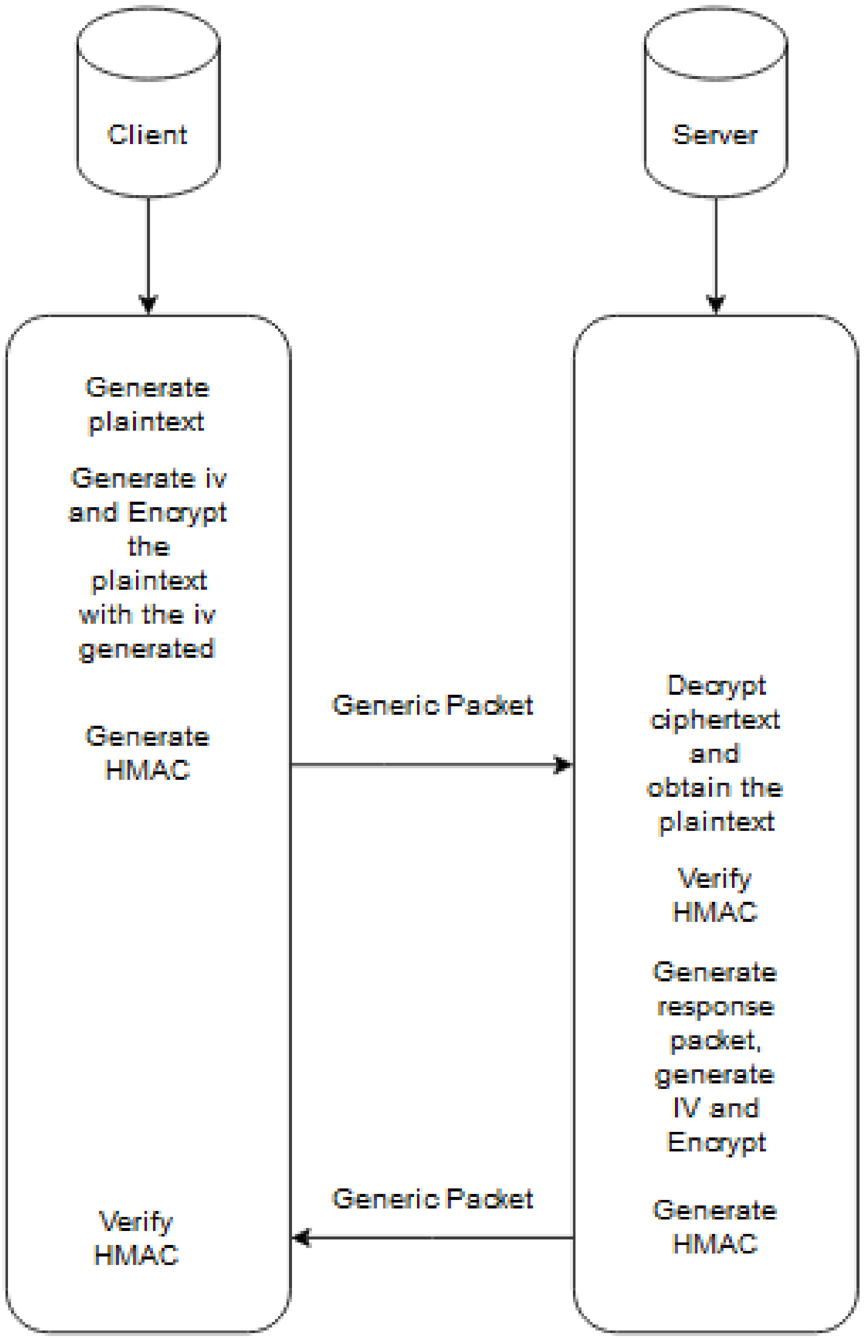
## 4.1 AUTHENTICATION

For the authentication between client and server, the workflow is managed using the startPacket and genericPacket packages.



## 4.2 COMMUNICATION

Generic packets are used for communication between server and client in a symmetric key.


Client
Server
Generate plaintext
Generate iv and Encrypt the plaintext with the iv generated
Generate HMAC
Generic Packet
Decrypt ciphertext and obtain the plaintext
Verify HMAC
Generate response packet, generate IV and Encrypt
Generate HMAC
Generic Packet
Verify HMAC