CS19B036 ATM Documentation

ATM Overview

The ATM supports the following options,

- Viewing account balance
- Withdrawal of cash
- Deposition of cash
- Transferring money to other accounts by cash or digital transfer
- Changing the PIN

The ATM has an encrypted database so that even if the database is hacked, the information will be of no use. The encryption has been done using SHA-256 hash for features that are accessed one way and for balance, AES encryption has been used because otherwise, it's not possible to retrieve and modify the balance. Also, even if the attacker manages to crack the balance, it will be of no use because everything else is encrypted.

Implementation

The entire implementation has been divided into 4 files and 1 database.

ATM

This file integrates all the modules. The `populateDummyData()` function generates random dummy data and writes it to the database. This function is for demonstrative purpose only.

BalanceEncryptionAndDecryption

This file contains the mechanism to encrypt and decrypt balance from the encrypted database. It contains a `secretKey` and a `salt` for which modifications are not enabled so as to not entertain any chance of penetration by the user.

Data.csv

This file is merely a lookup table for usage demonstration purposes and not an actual file attached to the ATM. The data here is in human readable format and any function that writes to this can be safely ignored because the bank is dependent on the encrypted database.

The columns of the database are,

PIN, Account Number, Phone Number, Current Balance and a status denoting whether the account is blocked or not.

EncryptedData.csv

SHA-256 hash has been used to encrypt the data and for balance, AES is used. This contains the same columns as the Data.csv file.

Hidden.java

This file handles all the critical applications of the ATM. It implements the following interface,

```
interface HiddenMechanism {
    boolean isCorrectPIN(String accountNumber, String PIN);

    double getBalance(String accountNumber);

    void addBalance(String accountNumber, double amount);

    void removeBalance(String accountNumber, double amount);

    void blockAccount(String accountNumber);

    void sendMessage(String account, int status);

    void changePIN(String accountNumber);
}
```

If the user enters their PIN incorrectly more than 4 times, their account is blocked and can only be unblocked after visiting a bank. All the updates of the transaction are sent to the user's phone through SMS. The `status` is determined and passed to the `sendMessage` function to vary the message as per the situtation.

OptionsScreen.java

This screen is shown perpetually and is used to convey the idea of an ATM display. All the critical functions are *not* handled by this file in any way. Rather, the modules calls the `Hidden.java` file to perform actions.