LOS EXAM REPORT

Name : Varun CN Roll Number : 251100690019

1 . Problem :

In today's digital world, online financial transactions are growing rapidly. However, with this growth comes a rise in cyber-fraud, unauthorized access, and malicious attempts to manipulate transactions.
The main challenge is to create a secure, traceable, and transparent transaction process — even in a simple, simulated environment.

To Overcome this, we designed a simple and secure transaction application that

- To check for unauthorized users.
- To Detect suspicious transactions.
- To maintain logs for all successful and failed transfers for auditing.

2. Objectives :

The main objective of this project is to develop a small, command-line financial simulation tool that performs secure money transfers using basic Linux commands such as grep, awk, and sed.

It mainly includes:

- User Authentication:
  Verify sender and receiver identities before any transfer.
- Balance Validation:
  Prevent transactions when the sender lacks sufficient balance.
- Cybersecurity Check:
  Detect unauthorized access or suspiciously large transactions and block them.
- Logging & Traceability:
  Maintain a clear record of all transaction attempts — successful or failed.

Practical Use of Linux Commands:
Demonstrate how text-processing commands can manage and secure data in flat files.

3.Approach

- Created two files — users.txt to store user details and transactions.txt to record all transactions.
- The script first displays all registered users with their balances.
- User enters sender ID, receiver ID, and the transfer amount.
- Using grep, the script checks if both IDs exist in the users list.
- If IDs are invalid, it stops the transaction for security reasons.
- If the amount exceeds ₹10,000, the script flags it as suspicious and blocks it.
- If the sender has insufficient balance, the transfer fails safely.
- For valid transactions, balances are updated using the sed command.
- All transactions (success or failure) are logged in transactions.txt.
- This ensures simple, secure, and transparent fund transfers between valid users.

4.Code Explanation

1. These lines define the filenames for user and transaction records.

2. Touch ensures the files exist before running the script.

3. Takes user input for sender, receiver, and amount to transfer.

4. grep searches for the sender and receiver IDs in users.txt.

5. If either user ID doesn't exist, the system detects it as an unauthorized attempt and logs it as FAILED_UNAUTHORIZED.

```
if [ -z "$sender_balance" ] || [ -z "$receiver_balance" ]; then echo "Security Alert: Unauthorized Access Attempt!" echo "$sender,$receiver,$amount,FAILED_UNAUTHORIZED" >> "$TRANS_FILE" return fi
```

6. Any transaction above ₹10,000 is considered suspicious and automatically blocked.

if ["$amount" -gt 10000]; then echo "Suspicious Activity Detected: Large transaction blocked!" echo "$sender,$receiver,$amount,FAILED_SUSPICIOUS" >> "$TRANS_FILE" return fi

7. Checks if the sender has enough balance before proceeding.

8. Calculates new balances after the transfer.

9. Appends transaction details to transactions.txt with a status of SUCCESS or FAILED depending on the result.

echo "$sender,$receiver,$amount,SUCCESS" >> "$TRANS_FILE"

10. Displays user data neatly using awk.

view_users() {

  awk -F',' '{printf "ID: %-5s | Name: %-10s | Balance: ₹%s\n", $1, $2, $3}' "$USERS_FILE"

}

11. Shows all transaction history with sender, receiver, amount, and status

view_transactions() {

  awk -F',' '{printf "From: %-5s | To: %-5s | Amount: ₹%-6s | Status: %s\n", $1, $2, $3, $4}' "$TRANS_FILE"

}

12. Execution

CODE

#!/usr/bin/bash

USERS_FILE="user.txt" TRANS_FILE="transaction.txt"

touch "$USERS_FILE" "$TRANS_FILE"

send_money() { echo "Initiate Secure Transaction" read -p "Enter Sender ID: " sender read -p "Enter Receiver ID: " receiver read -p "Enter Amount to Transfer: " amount

sender_balance=$(grep "^$sender," "$USERS_FILE" | cut -d',' -f3) receiver_balance=$(grep "^$receiver," "$USERS_FILE" | cut -d',' -f3)

Cybersecurity Check

if [-z "$sender_balance"] || [-z "$receiver_balance"]; then echo "Security Alert: Unauthorized Access Attempt!" echo "$sender,$receiver,$amount,FAILED_UNAUTHORIZED" >> "$TRANS_FILE" return fi

Detect suspicious large transaction

if ["$amount" -gt 10000]; then echo "Suspicious Activity Detected: Large transaction blocked!" echo "$sender,$receiver,$amount,FAILED_SUSPICIOUS" >> "$TRANS_FILE" return fi

Check balance

if ["$amount" -le "$sender_balance"]; then new_sender_balance=$((sender_balance - amount)) new_receiver_balance=$((receiver_balance + amount))

```
# Update balances securely
sed -i "s/^$sender,[^,]*,[0-9]*/$sender,$(grep "^$sender," 
"$USERS_FILE" | cut -d',' -f2),$new_sender_balance/" "$USERS_FILE"
sed -i "s/^$receiver,[^,]*,[0-9]*/$receiver,$(grep "^$receiver," 
"$USERS_FILE" | cut -d',' -f2),$new_receiver_balance/" "$USERS_FILE"

echo "Transaction Successful!"
echo "$sender,$receiver,$amount,SUCCESS" >> "$TRANS_FILE"
```

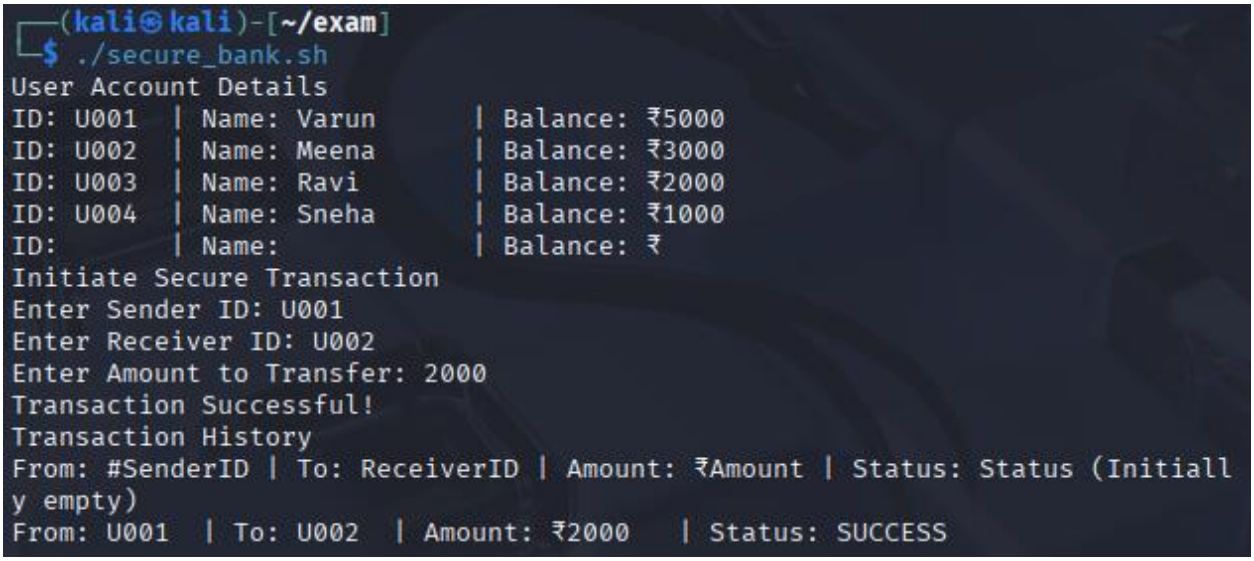
else echo " Transaction Failed: Insufficient Funds!" echo "$sender,$receiver,$amount,FAILED_LOW_BALANCE" >> "$TRANS_FILE" fi }

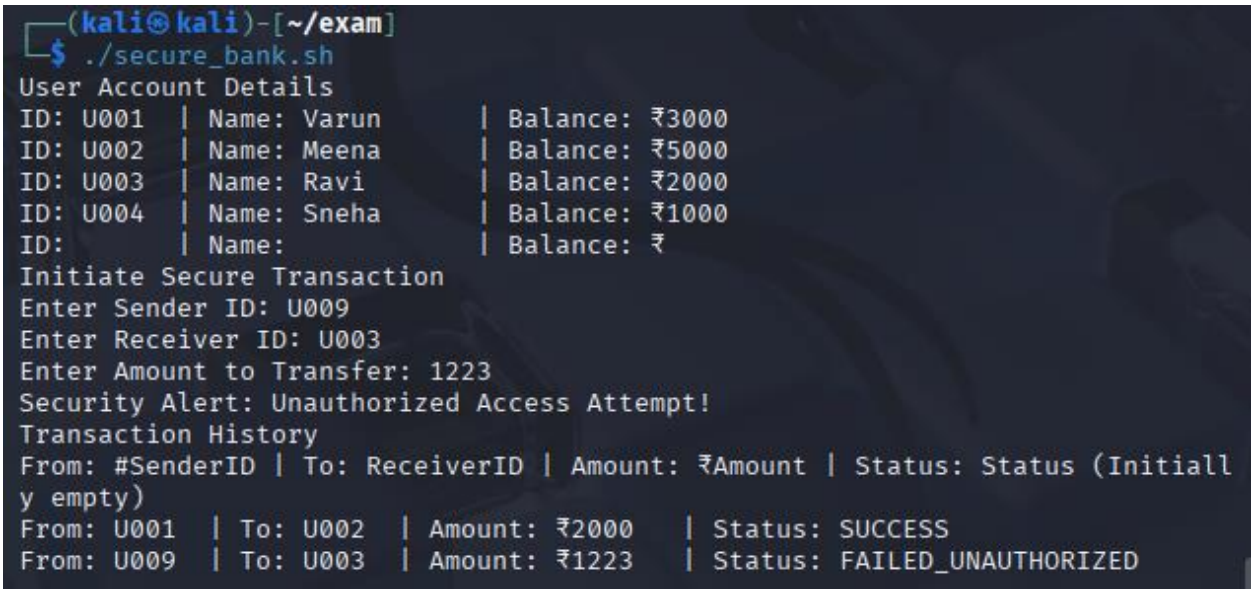
view_users() { echo "User Account Details" awk -F',' '{printf "ID: %-5s | Name: %-10s | Balance: ₹%s\n", $1, $2, $3}' "$USERS_FILE" }

view_transactions() { echo "Transaction History" awk -F',' '{printf "From: %-5s | To: %-5s | Amount: ₹%-6s | Status: %s\n", $1, $2, $3, $4}' "$TRANS_FILE" }

#SAMPLE RUN view_users send_money view_transactions

SCREENSHOT


(kali㉿kali)-[~/exam]
$ ./secure_bank.sh
User Account Details
ID: U001 | Name: Varun | Balance: ₹5000
ID: U002 | Name: Meena | Balance: ₹3000
ID: U003 | Name: Ravi | Balance: ₹2000
ID: U004 | Name: Sneha | Balance: ₹1000
ID: | Name: | Balance: ₹
Initiate Secure Transaction
Enter Sender ID: U001
Enter Receiver ID: U002
Enter Amount to Transfer: 2000
Transaction Successful!
Transaction History
From: #SenderID | To: ReceiverID | Amount: ₹Amount | Status: Status (Initially empty)
From: U001 | To: U002 | Amount: ₹2000 | Status: SUCCESS



LINK

[cnvarun19-cmyk/LOS-REPORT]