CSU34031 Project 2

Student name: Davy Nolan
Student number: 17330208

Introduction

The objective of this project is "to develop a secure social media application for Facebook, **Twitter**, WhatsApp etc., or for your own social networking app. For example, your application will secure the Facebook Wall, such that only people that are part of your "Secure Facebook Group" will be able to decrypt each other's posts. To all other users of the system the post will appear as ciphertext".

Requirements

- ⇒ Design and implement a suitable key management system for your application that allows any member of the group to share social media messages securely.
- ⇒ Allows you to add or remove people from a group.
- ⇒ Free to implement application for a desktop or mobile platform and make use of any open source cryptographic libraries.

Implementation

I decided to create a python application using **Tweepy** to access the Twitter API. This application allows the user to:

1. View the home timeline of their Twitter.
2. View another Twitter user's timeline.
3. Post a tweet on the timeline.
4. View the list of users that are members of the Secure Group (i.e. Users that can view each other's content)
5. Add users to the Secure Group.
6. Remove users from the Secure Group.

Encryption/Decryption and Key Management

I made use of the python library **cryptography** and **Fernet** to allow me to encrypt and decrypt text from tweets.
Fernet is used to create a key which is used when encrypting and decrypting text data. The function **create_encrypt_key()** creates a new key and stores it in a file called **key.key**. The function **read_key()** then reads the previously created key from **key.key** returns it in bytes.
Every time the application is run; a new key is created and **key.key** is overwritten with the new key. This ensures maximum security of encrypted data.

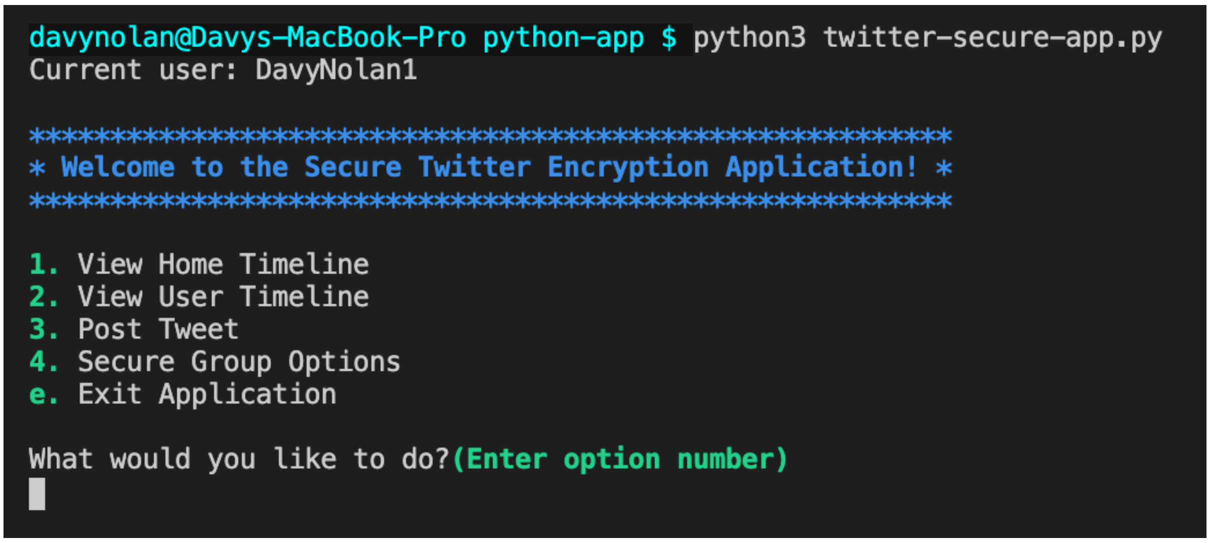
The function **encrypt_tweet(tweet)** takes a Tweet object as an argument and returns the encrypted version of the text from the Tweet using the key saved previously.

Group Membership

Group members are recognised by their Twitter usernames in this system. This program is accompanied by a file called **user-group.txt** which contains a list of usernames which are members of the Secure Group. Members of the Secure Group can decrypt each other's tweets.
Let userB be a user who is a member of the Secure Group and let userA be a member who is **not** a member of the Secure Group. Whenever userA uses this application to try access userB's tweets, they will appear as encrypted text. UserA must be a member of the Secure Group in order to view userB's tweets.
Let userC be a user who is **not** a member of the Secure Group. Whenever userA uses this application to try access userC's tweets, they will appear as decrypted text.
The function **encrypt_check(tweet)** extracts the username from the tweet passed to the function and then checks to see if that user is in the Secure Group.
If the extracted username is in the Secure Group but the user of the application is not in the Secure Group, the encrypted text of the tweet is returned – this is done by using the **encrypt_tweet(tweet)** function.
Otherwise the decrypted text of the tweet is returned.

User Interface

Running **python3 twitter-secure-app.py** will open up the UI for the program. You will be met by 5 options: Entering in any option from the list will perform the corresponding function explained.



```
davynolan@Davys-MacBook-Pro python-app $ python3 twitter-secure-app.py
Current user: DavyNolan1

*********************************************************
* Welcome to the Secure Twitter Encryption Application! *
*********************************************************

1. View Home Timeline
2. View User Timeline
3. Post Tweet
4. Secure Group Options
e. Exit Application

What would you like to do?(Enter option number)
```

1. View Home Timeline

Prints the 20 most recent tweets on the application user's timeline. It does this by calling the **display_home_tl()** function which calls the Twitter API to get the tweets on the user's home timeline

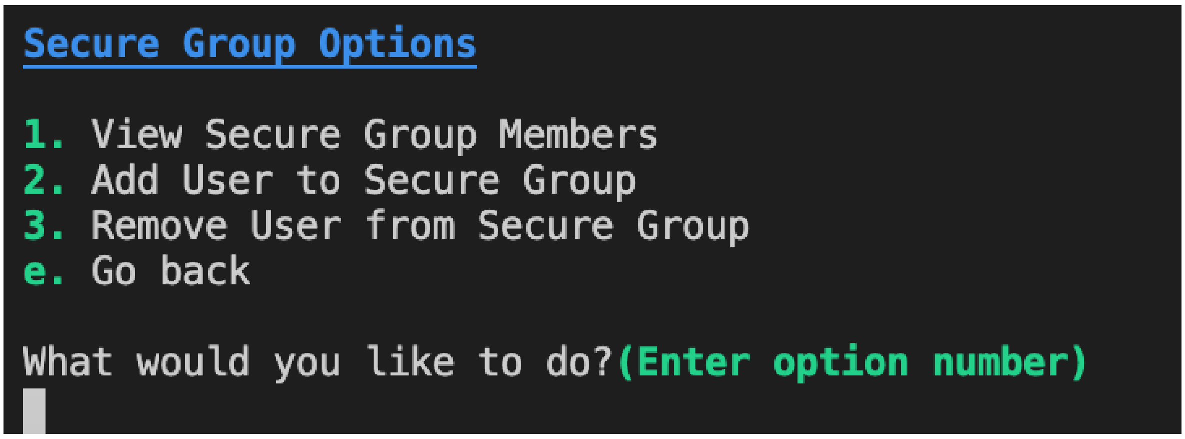
2. View User Timeline

Asks the user to enter in a username and then prints the 20 most recent tweets made by that user. It does this by calling the **display_user_tl(username)** function which calls the Twitter API to get the tweets of the specified user.

3. Post Tweet

Asks the user to enter in the text of the tweet they wish to post and then posts the new tweet to the timeline. It does this by calling the **tweet_text(text)** function which calls the Twitter API to post a tweet.

4. Secure Group Options



```
Secure Group Options

1. View Secure Group Members
2. Add User to Secure Group
3. Remove User from Secure Group
e. Go back

What would you like to do?(Enter option number)
```

This opens up a UI for the Secure Group Options. These allow you to view what usernames are currently in the Secure Group and to add/remove users from the group.

Set-up

Before running the application, you must create a Twitter developer account and wait for approval from Twitter. Once it is approved, you must provide your **Consumer_key** , **Consumer_secret, key** and **secret** API keys. You must enter these keys into the accompanied file **api_keys.py**.
This application uses the python libraries **Tweepy** and **Cryptography** so these must be installed before use.

Video Demo

My video demonstration of the application can be found here.

Github

Application can be cloned and downloaded from here.

Code Appendix

```
import json

import sys

import cryptography

from cryptography.fernet import Fernet

from api_keys import CONSUMER_KEY, CONSUMER_SECRET, KEY, SECRET

import tweepy


f = open("user-group.txt", "r")

secure_group = f.read().splitlines()

try:

   auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)

   auth.set_access_token(KEY, SECRET)

   api = tweepy.API(auth)

   current_user = api.me().screen_name

except:

   print("Error: Please enter your API keys into api_keys.py file.")

   exit(0)




def secure_user(username):

   if username in secure_group:

      return True

   return False


def create_encrypt_key():

   key = Fernet.generate_key()

   file = open('key.key', 'wb')

   file.write(key)

   file.close()


def read_key():

   file = open('key.key', 'rb')

   key = file.read()

   file.close()

   return key


def encrypt_tweet(tweet):
   text = tweet.text.encode()
   f = Fernet(read_key())
   return f.encrypt(text)

def encrypt_check(tweet):
   print(bcolors.FAIL+"@"+bcolors.BOLD+bcolors.OKBLUE+tweet.user.screen_name+bcolors.ENDC+":")
   if(secure_user(tweet.user.screen_name) and not(secure_user(current_user))):
      return encrypt_tweet(tweet)
   return tweet.text

def display_user_tl(username):
   print(username)
   try:
      for tweet in api.user_timeline(username):
          print(encrypt_check(tweet))
          print()
   except:
      print(bcolors.BOLD+bcolors.FAIL+"User does not exist."+bcolors.ENDC)

def display_home_tl():
   for tweet in api.home_timeline():
      print(encrypt_check(tweet))
      print()

def tweet_text(text):
   try:
      api.update_status(text)
      print(bcolors.BOLD+bcolors.OKGREEN+"Tweet successfully posted."+bcolors.ENDC)
   except:
      print(bcolors.BOLD+bcolors.FAIL+"Tweet failed to post."+bcolors.ENDC)



def update_secure_group():
   f = open("user-group.txt", "r")
   secure_group = f.read().splitlines()

def display_sec_group():
   print(bcolors.BOLD + bcolors.UNDERLINE + bcolors.OKBLUE+"\nSecure Group Members:"+bcolors.ENDC)
   for user in secure_group:
      print(user)
```

```
def add_to_sec_group(username):
    filename = "user-group.txt"
    if secure_user(username):
        print(bcolors.BOLD +bcolors.FAIL+username+" is already in Secure Group."+ bcolors.ENDC)
        return
    bl= open(filename,'a')
    try:
        bl.write(username+"\n")
        print(bcolors.BOLD +bcolors.OKGREEN+"Successfully added "+username+" to Secure Group."+
bcolors.ENDC)
    except:
        print(bcolors.BOLD +bcolors.FAIL+"Failed to add "+username+" to Secure Group."+ bcolors.ENDC)
    bl.close()
    update_secure_group()

def rem_from_sec_group(username):
    filename = "user-group.txt"
    is_in_list = 0
    try:
        with open(filename, "r") as f:
            lines = f.readlines()
    except:
        print(bcolors.BOLD +bcolors.FAIL+"Error: Could not open "+filename+"."+ bcolors.ENDC)

    try:
        with open(filename, "w") as f:
            for line in lines:
                if line.strip("\n") != username:
                    f.write(line)
                if line.strip("\n") == username:
                    is_in_list =1
        f.close()
    except:
        print(bcolors.BOLD +bcolors.FAIL+"Error: Could not open "+filename+"."+ bcolors.ENDC)
    if(is_in_list == 0):
        print(bcolors.BOLD +bcolors.FAIL+username, "is not in Secure Group so cannot be
removed."+bcolors.ENDC)
    elif(is_in_list == 1):
        print(bcolors.BOLD +bcolors.OKGREEN+ username,"successfully removed from Secure Group"+
bcolors.ENDC)

    update_secure_group()


class bcolors:
    HEADER = '\033[95m'
    OKBLUE = '\033[94m'
    OKGREEN = '\033[92m'
    WARNING = '\033[93m'
    FAIL = '\033[91m'
    ENDC = '\033[0m'
    BOLD = '\033[1m'
    UNDERLINE = '\033[4m'

def ui_main_menu():
    print(bcolors.BOLD + bcolors.OKBLUE+"\n************************************************************"+
bcolors.ENDC)
    print(bcolors.BOLD + bcolors.OKBLUE+"* Welcome to the Secure Twitter Encryption Application! *"+
bcolors.ENDC)
    print(bcolors.BOLD + bcolors.OKBLUE+"************************************************************"+ bcolors.ENDC)

    print("\n"+bcolors.BOLD +bcolors.OKGREEN+"1."+bcolors.ENDC+" View Home Timeline")
    print(bcolors.BOLD +bcolors.OKGREEN+"2."+bcolors.ENDC+" View User Timeline")
    print(bcolors.BOLD +bcolors.OKGREEN+"3."+bcolors.ENDC+" Post Tweet")
    print(bcolors.BOLD +bcolors.OKGREEN+"4."+bcolors.ENDC+" Secure Group Options")
    print(bcolors.BOLD +bcolors.OKGREEN+"e."+bcolors.ENDC+" Exit Application\n")

    try:
        option = input("What would you like to do?"+ bcolors.BOLD +bcolors.OKGREEN+"(Enter option
number)\n"+bcolors.ENDC)

        if(option == '1'):
            display_home_tl()
            go_back = input(bcolors.BOLD +bcolors.OKGREEN+"Go back to main menu? (y/n): "+bcolors.ENDC)
            if(go_back == 'y'):
                ui_main_menu()
            else:
                print(bcolors.BOLD+bcolors.OKGREEN+"\nGoodbye."+bcolors.ENDC)
                return

        if(option == '2'):
            username = input(bcolors.BOLD +"Enter username to view their timeline: "+ bcolors.ENDC)
            display_user_tl(username)
            go_back = input(bcolors.BOLD +bcolors.OKGREEN+"Go back to main menu? (y/n): "+bcolors.ENDC)
            if(go_back == 'y'):
                ui_main_menu()
            else:
                print(bcolors.BOLD+bcolors.OKGREEN+"\nGoodbye."+bcolors.ENDC)
                return

        if(option == '3'):
            text = input(bcolors.BOLD +"Enter text of tweet you wish to post: "+ bcolors.ENDC)
            tweet_text(text)
            go_back = input(bcolors.BOLD +bcolors.OKGREEN+"Go back to main menu? (y/n): "+bcolors.ENDC)
            if(go_back == 'y'):
                ui_main_menu()
            else:
                print(bcolors.BOLD+bcolors.OKGREEN+"\nGoodbye."+bcolors.ENDC)
                return

        if(option == '4'):
            ui_sec_group()

        if(option == 'e'):
            print(bcolors.BOLD+bcolors.OKGREEN+"\nGoodbye."+bcolors.ENDC)
            return

    except KeyboardInterrupt:
        print(bcolors.BOLD+bcolors.OKGREEN+"\nGoodbye."+bcolors.ENDC)

def ui_sec_group():
    if secure_user(current_user):
        print(bcolors.BOLD + bcolors.UNDERLINE + bcolors.OKBLUE+"\nSecure Group Options"+ bcolors.ENDC)
        print("\n"+bcolors.BOLD +bcolors.OKGREEN+"1."+bcolors.ENDC+" View Secure Group Members")
        print(bcolors.BOLD +bcolors.OKGREEN+"2."+bcolors.ENDC+" Add User to Secure Group")
        print(bcolors.BOLD +bcolors.OKGREEN+"3."+bcolors.ENDC+" Remove User from Secure Group")
        print(bcolors.BOLD +bcolors.OKGREEN+"e."+bcolors.ENDC+" Go back\n")
        try:
            option = input("What would you like to do?"+ bcolors.BOLD +bcolors.OKGREEN+"(Enter option
number)\n"+bcolors.ENDC)

            if(option == '1'):
                display_sec_group()

            if(option == '2'):
                display_sec_group()
                username = input(bcolors.BOLD +"Enter username you wish to add to Secure Group: "+
bcolors.ENDC)
                add_to_sec_group(username)

            if(option == '3'):
                display_sec_group()
                username = input(bcolors.BOLD +"Enter username you wish to remove from Secure Group: "+
bcolors.ENDC)
                rem_from_sec_group(username)

            if(option == 'e'):
                ui_main_menu()

        except KeyboardInterrupt:
            print (bcolors.BOLD+bcolors.OKGREEN+"\nGoodbye."+bcolors.ENDC)
    else:
        print(bcolors.BOLD +bcolors.FAIL+"Access restriced; current user is not in Secure Group."+bcolors.ENDC)




if __name__ == "__main__":
    create_encrypt_key()
    print("Current user: " + current_user)
    if(len(secure_group) == 0):
        add_to_sec_group(current_user)
    ui_main_menu()
```