Immersion Cybersecurity (CTF)

Bonus: ft_otp

Summary: *Nothing ever lasts forever...*

Version: 1.00

Contents

Chapter I

Introduction

Passwords are `one of the biggest headaches in computer security`. Users forget them, share them, reuse them, and choose them horribly. Furthermore, passwords are sooner or later leaked in security breaches. One way to avoid this is to use one-time passwords, based on timestamps, which expire after a few minutes and then become invalid. Whether you already use this system, or if you have never heard of it, it is quite likely that one of your passwords has been compromised at some point in your life.

In this project, the aim is `to implement a TOTP` (Time-based One-Time Password) system, which will be capable of generating ephemeral passwords from a master key. It will be based on the `RFC`: https://datatracker.ietf.org/doc/html/rfc6238, So you could use it in your day-to-day.

Chapter II

Prologue

The Silk Road stretched across the entire Asian continent, connecting China with Mongolia, Persia, India, the Middle East, Turkey, Europe, and Africa. Despite the name, it was not the valuable cloth that was primarily traded. Glass, leather, weapons, or war machines traveled the world, expanding industrial discoveries, printing techniques, gunpowder, or the compass.

Password, please?
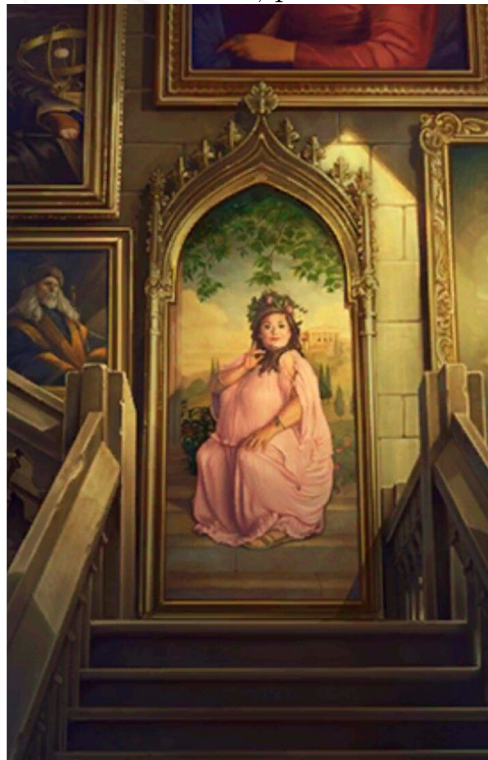
Chapter III

Mandatory Part

In the language of your choice, you have to implement a program that allows you to store an initial password in a file and is capable of generating a new one-time password every time it is requested. You can use any library that facilitates the implementation of the algorithm, as long as it doesn't do the dirty work, i.e., using a TOTP library is strictly prohibited. Of course, you can and should use a library or function that allows you to access system time.

- The executable must be named `ft_otp`.

- Your program must take arguments.

  - `-g`: The program receives a hexadecimal key of at least 64 characters as an argument. The program securely stores this key in a file called `ft_otp.key`, which is encrypted.

  - `-k`: The program generates a new temporary password based on the key given as an argument and prints it to the standard output.

- Your program must use the HOTP algorithm (RFC 4226).

- The generated one-time password must be random and always contain the same format, i.e., 6 digits.

  Below is an example of use:

```
$ echo -n "NEVER GONNA GIVE YOU UP" > key.txt
$ ./ft\_otp -g key.txt
./ft\_otp: error: key must be 64 hexadecimal characters.
$ [..]
$ cat key.hex | wc -c
64
$ ./ft\_otp -g key.hex
Key was successfully saved in ft\_otp.key.
$ ./ft\_otp -k ft\_otp.key
836492
$ sleep 60
$ ./ft\_otp -k ft\_otp.key
123518
```

You can check if your program is working properly by comparing generated passwords with `Oathtool` or any tool of your choice.

```
oathtool -totp $(cat key.hex)
```

Please note that the reference software you choose will be used
during your evaluation.

Chapter IV

Bonus Part

You can enhance your project with the following features:

- Creation of a QR code with seed generation.

- Creation of a graphic interface.

The bonus part will only be evaluated if the mandatory part is PERFECT. "Perfect" means the mandatory part has been entirely completed and works without malfunctioning. If you have not fulfilled ALL the mandatory requirements, your bonus part will not be evaluated at all.

Chapter V

Submission and peer-evaluation

- Create a new 'bonus00-ft_otp' folder and navigate to it. Place the files inside the folder and then push it.