INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL
DELEGAÇÃO REGIONAL DE LISBOA E VALE DO TEJO
CENTRO DE EMPREGO E FORMAÇÃO PROFISSIONAL DE LISBOA

Técnico de Redes – Sistemas – CET-TEGRSI08

Projeto 2 – Pycracker e Syms

Por: Rafael Oliveira Nº20 e Hugo Pinto Nº8

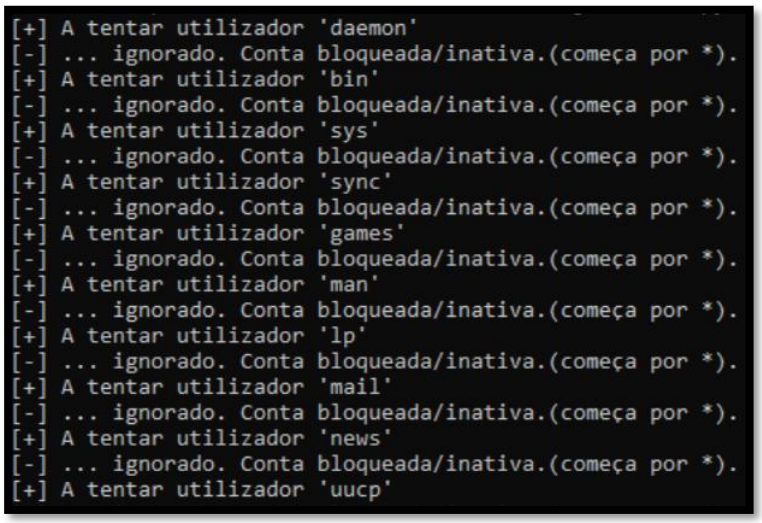

Figure 1 - Verbose em Pycracker

```
(.venv) 21:53:37 ~/Desktop/Projeto_Pycracker_Syms/src $ syms.py -h
Returns all duplciate files in the given folder.

Usage:
    ./syms.py [-c] [-n] [-e] [-r PATTERN] [DIR_PATH]

Options:
    DIR_PATH                        Start directory [defalt: .]
    -c, --contents                  Search files with the same binary content
    -n, --name                      Search files with the same name
    -e, --extension                 Search files with the same extension
    -r PATTERN, --regex==PATTERN    Search files using a regular expression
```

Figure 2 - Ajuda na invocação do Syms.py

Índice

Introdução e Objetivos

Este projeto tem como objetivo testar e aprimorar os nossos conhecimentos em Python adquiridos até ao momento, através de dois programas.

Pycracker que recebe e manipula os documentos disponibilizados para achar e "crackar" as palavras-chave dos utilizadores.

E Syms, um programa que recebe um caminho até á directoria desejada e dispõe os arquivos que contenham certas similaridades, como nome ou extensão.

```
(.venv) 22:10:30 ~/Desktop/Projeto_Pycracker_Syms/src $ syms.py -n ../tests
_______________BY NAME_____________________
Restaurante1.cpp
   ../tests/duplicados/Restaurante1.cpp
   ../tests/duplicados/dir1/subdir1/Restaurante1.cpp

Lab.pdf
   ../tests/duplicados/Lab.pdf
   ../tests/duplicados/dir2/Lab.pdf

FahrCelsius2.cs
   ../tests/duplicados/FahrCelsius2.cs
   ../tests/duplicados/dir2/FahrCelsius2.cs

documento.pdf
   ../tests/duplicados/documento.pdf
   ../tests/duplicados/dir2/documento.pdf

._documento.pdf
   ../tests/duplicados/._documento.pdf
   ../tests/duplicados/dir2/._documento.pdf

-----------------------------------------------------
```

Figure 3 - Invocação do programa Syms organizando pelo nome.

Análise

Em sistemas Linux, as palavras-chave são encriptadas (hashes) e armazenadas no ficheiro "/etc/shadow", juntamente com informações adicionais, como a data de expiração e as politicas de palavra-chave.

Este é um exemplo de um registro no ficheiro "shadow":

"username:hash:ultima_mudança…expiração_conta:reservado"

Os campos relevantes neste caso são o "username" ou nome de utilizador e o "hash" a palavra-chave encriptada.

O algoritmo mais comum de "hashing" é o SHA-512, embora outros, como MD5 e SHA-256, possam ser usados em sistemas mais antigos.

Desenho e Estrutura

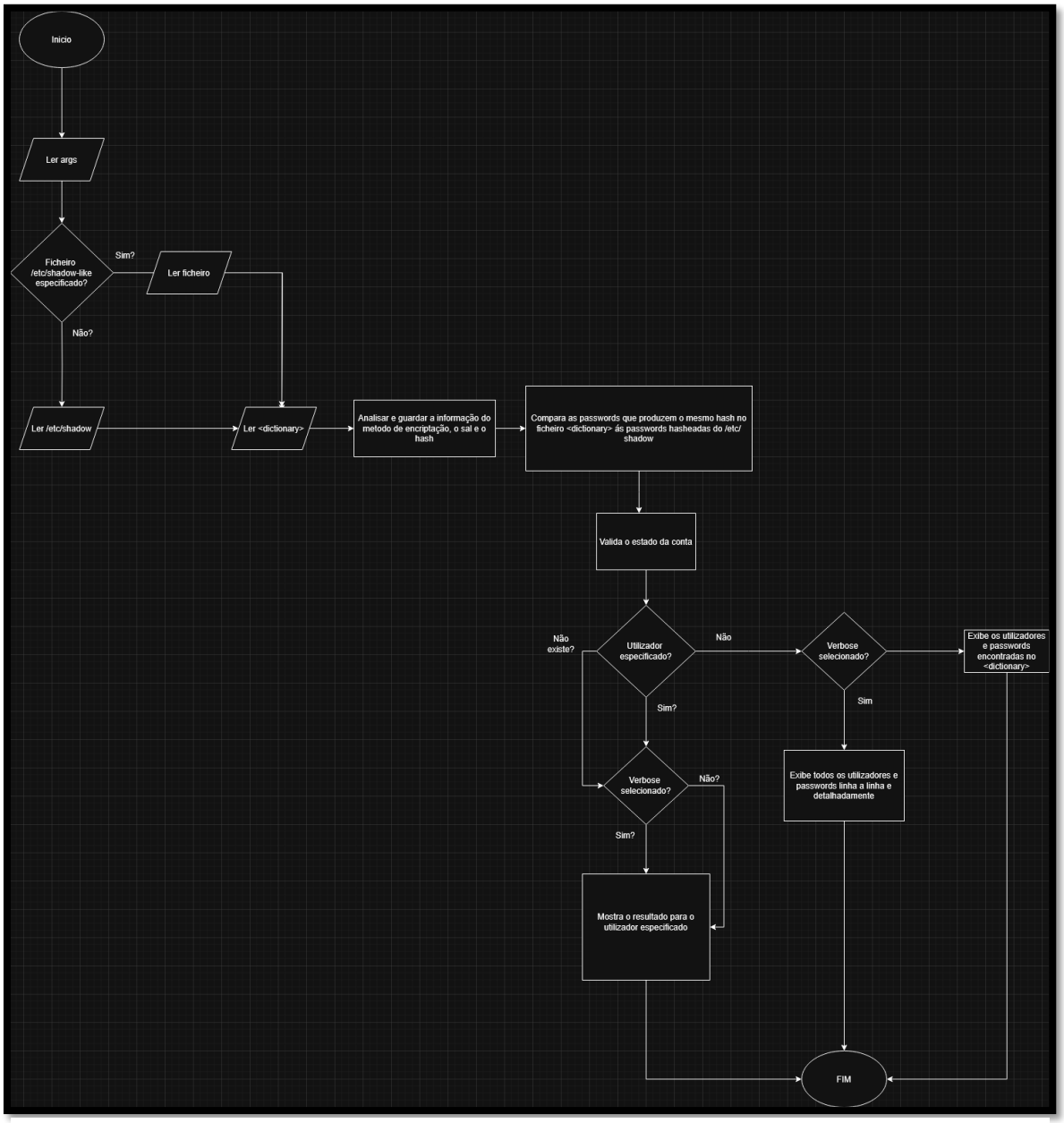

Figure 4 - Fluxograma do PyCracker

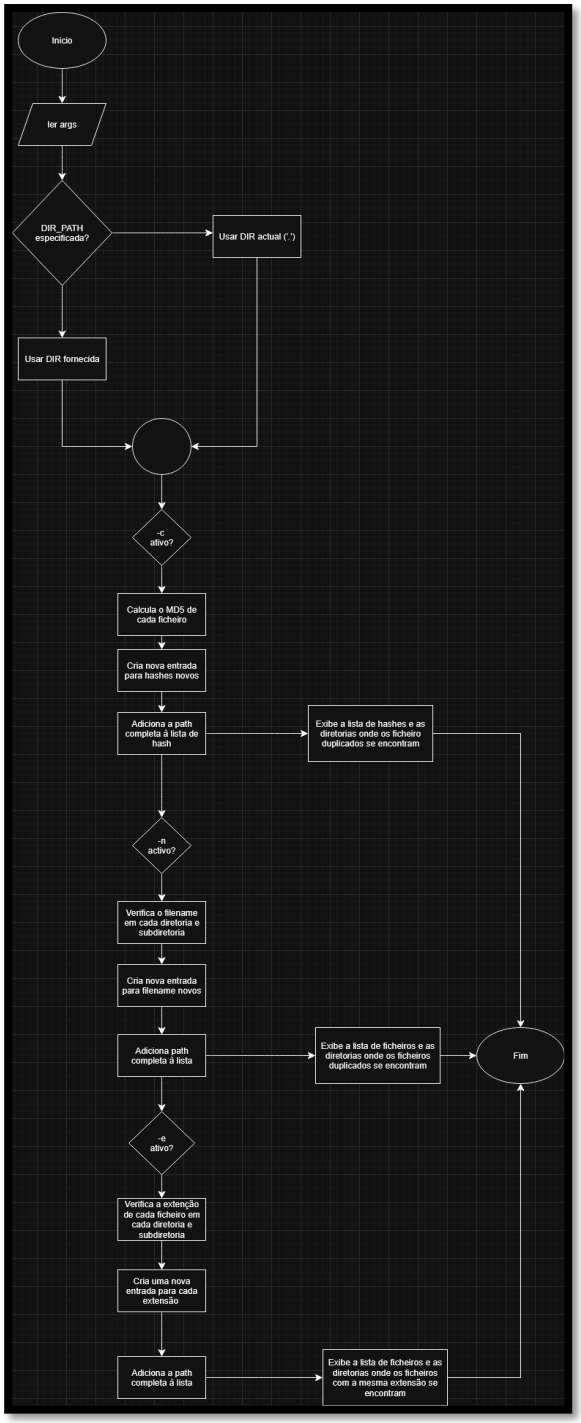

Figure 5 - Fluxograma do Syms

Implementação

Durante o Syms, as estruturas de dados principais foram os dicionários (dict) e as listas (lists) para armazenar grupos de ficheiros duplicados e caminhos de ficheiros que correspondem á expressão regular fornecida.

Já no caso dos módulos, foram utilizados: sys, docopt, os, hashlib e re para acessar a argumentos na linha de comandos, analisar argumentos com base numa string de documentação, interagir com o sistema de ficheiros para navegar diretorias e construir caminhos, gerar hashes para comparar conteúdos de ficheiros e realizar a correspondência de padrões em nomes de ficheiros usando expressões regulares.

Já no PyCracker, as estrutudas de dados principais foram os dicionários (dict), as listas (lists) e os tuplos (tuple) para armazenar correspondências entre utilizadores e palavras-chave decifradas, junto com o método de hash e para iterar palavras-chave do ficheiro do dicionário.

Os módulos utilizados neste programa foram: sys, docopt, enum, typing, textwrap, passlib e argparse, para acessar e analisar argumentos da linha de comandos, adicionar anotações de tipo para melhorar a legibilidade e validação do código, formatar a string de documentação para docopt, verificar e manipular hashes e como o argparse como alternativa ao docopt.

Conclusão

Apesar de desafiador, foi um projeto onde aprendemos bastante sobre os diversos aspetos da linguagem de programação Python.

Durante o desenvolvimento do programa fomos capazes de recriar o código presente nas gravações do formador e cumprir os desafios propostos.

Agradecemos ao formador João Galamba pela oportunidade de não só evoluir neste aspeto como preparar um projeto que seja condizente com as nossas capacidades. Obrigado!