Security 2023 Assignment 1
Bruteforcing

Daniël Zee s2063131

March 2023

1 Problem description

We are given two files: a zip archive and a txt file. The zip archive is password protected and contains another password protected archive. These nested archives go on until a hidden *flag* is found. The txt file contains a big list of passwords to try on the zip files. The goal of the assignment was to write a python script to automate the extraction of the nested zip files by trying all the passwords in the txt file until the *flag* had been found.

2 Solution implementation

The script first needs to know the path of the zip archive and the txt file. Using the *glob* package we can get a list of all the files in a directory with a specific file extension (.zip and .txt). Because we expect only a single zip file and txt file we assume the first one we find to be the one we need.

We then go through the txt file line by line and try every one as a password to try to decrypt and extract the the archive using the *zipfile* package. This package raises errors when the passwords are incorrect so we catch those and continue to the next line in the txt file. We raise an error ourselves if all the passwords in the txt were incorrect.

When the script has found the correct password for an archive it will extract the contents to a directory with the same name as the archive (for example the archive *rlpph.zip* gets extracted to the directory *rlpph*). We then remember the path of this directory and start the process of looking for zip archives again in this directory (the txt file for the passwords stays the same of course). When a zip archive has been found the whole process of trying to decrypt and extract it starts over again. This results in every archive being extracted in a subdirectory of the directory of the previously extracted archive (for example *rlpph/hgxmn/olizo/*).

This cycle ends when the last extracted archive does not contain another archive but some other file. We assume that this file is our *flag* we are looking for. The script prints the content of this file to the terminal and moves it to same directory as the first archive and the txt file. We then neatly clean up our work by removing the directory of the first archive (which contains the directories of all the other extracted archives).

The script does not expect arguments to work but the user has the option to show the progress and the correct passwords for every archive by providing the *-v* argument.