Introduction to Cyber Security

What is our GOAL for this CLASS?

In this class, we have learned about Cybersecurity and will try to gain access to password protected files

What did we ACHIEVE in the class TODAY?

- Understand about Cyber Security
- Using brute force, retrieve the password for the PDF file
- Using brute force attack to recover the password for protected zip files

Which CONCEPTS/ CODING BLOCKS did we cover today?

- We used the libraries, PyPDF2, zipfile, time
- We used methods pdfReader.decrypt(word),zf.extractall

Understanding concepts:

Cyber Security:

Cyber security refers to the technologies, processes, and practices designed to protect networks, devices, programs, and data from attack, damage, or unauthorized access.

Brute Force Attack:

A brute force attack uses trial-and-error to guess user passwords. In brute force attack, a list of commonly used passwords is used against a user account or protected documents, such as 123456, password123, qwerty, abc123 or it runs an algorithm against an encrypted password.

How did we DO the activities?

1. Create an encrypted folder: Steps to set a password for the folder
    - Create a new folder and insert a text file, a pdf file, or an image on which you will set a password, but make sure that the folder is not empty
    - Right-click on the folder and add to the archive by clicking on zip
    - Select Archive format zip on the left top side
    - On the right side, there is an Encryption option, write down a 4-digit password using numbers and small alphabets only

2. The **zipfile** module **:**Python has an inbuilt module named **zipfile** that can be used to access zip files. Open the terminal and install the dependencies

    - Import **zipfile**
    - Import **time** to check the **time** to decode the password
    - Get the folder path using **input**
    - Initialize **zipfile** object

```
import zipfile
import time

folderpath = input('Path to the file: ')
zipf = zipfile.ZipFile(folderpath)
```

    - Write the if statement to check if the folder is password protected or not, if not then print the message.
    - Else start the timer, initialize a variable result with **'0'** will indicate **Failure**, while **'1'** will indicate **Success**
    - Initialize a variable **c** to keep the count of passwords tried
    - Build a character array including all numbers, lowercase letters
    - Print "Brute Force started"

```
characters =['0','1','2','3','4','5','6','7','8','9',
             'a','b','c','d','e','f','g','h','i','j','l','k','m','n','o','p','q','r','s','t','u','v','w','x','y','z']


print("Brute Force Started...")
```

```
if not zipf:
    print('The zipped file/folder is not password protected! You can successfully open it!')

else:
    starttime = time.time()
    result = 0
    c = 0
```

3. Logic to **generate four digit password** and check the result:
    - To generate four digit password
        - **“i”** loop is used to get **first** character from array and
        - then it will make possible combination with **“j”** loop,
        - **“k”** loop and
        - **“l”** loop
    - After getting all possible combinations it will save all the four digit-character combinations into guess variable
    - Print the **guess**
    - Increment the variable **c** by 1
    - Now in try use **“Open”** a **ZIP** file, where file can be a path to a file
    - The mode parameter should be **“ r “** to read an existing file, **“zf.extractall”** members from the archive to the current working directory.
    - **pwd** is the password used for encrypted files.
    - After getting the correct password print **“Success”** Message and stop the timer and result variable set to be **1** as its **true** condition.
    - In case of **success**, **break** the **“i”** loop
    - In case of an **exception**, **pass** it.
    - If the **password** is found, break from **”j”** for loop, similarly **“k”** and **“l”** loop too.

```
if(result == 0):
    print("Checking for 4 character password...")
    for i in characters:
        for j in characters:
            for k in characters:
                for l in characters:
                    guess = str(i) + str(j) + str(k) + str(l)
                    password=guess.encode('utf8').strip()
                    print(guess)
                    c=c+1
                    try:
                        with zipfile.ZipFile(folderpath,'r') as zf:
                            zf.extractall(pwd=password)
                            print("Success! The password is: "+ guess)
                            endtime = time.time()            #Save the end time
                            result = 1                       #Set result variable to 1 on success
                            break                            #If the password is found break from i for loop
                    except:
                        pass
                if result == 1:
                    break                                #If the password is found break from j for loop
            if result == 1:
                break                                    #If the password is found break from k for loop
        if result == 1:
            break                                        #If the password is found break from l for loop
```

● If no four-character password is found, print the password not found along with the time and number of times it was tried.
● Else password found, print the duration and display the number of times the password has been tried along with congratulation message

```
if(result == 0):
    print("Sorry, password not found. A total of "+str(c)+" possible combinations tried in "+str(duration)+" seconds. Password is not of 4 characters.")
else:
    duration = endtime - starttime
    print('Congratulations!!! Password found after trying '+str(c)+' combinations in '+str(duration)+' seconds')
```

```
Path to the file: C:\Users\User\Pictures\Brute Force Zip Folder\0004.zip
Brute Force Started...
Checking for 4 character password...
0000
0001
0002
0003
0004
Success! The password is: 0004
Congratulations!!! Password found after trying 5 combinations in 0.022717714309
92383 seconds
```

4. Import **PyPDF2** library for pdf reader

   - Get the folder path using input
   - By default, the **open()** function opens a file in text format. Add to the mode parameter to open a binary file. As a result, the mode opens the file in binary format for reading
   - Using if statement check if the pdf is **password** protected or not, if not then print the message

```
import PyPDF2 as pd
filename = input('Path to the file: ')
file = open(filename,'rb')
pdfReader = pd.PdfFileReader(file)
```

```
if not pdfReader.isEncrypted:
    print('The file is not password protected! You can successfully open it!')
```

```
else:
    wordListFile = open('wordlist.txt', 'r',errors='ignore')
    body = wordListFile.read().lower()
    words = body.split('\n')
```

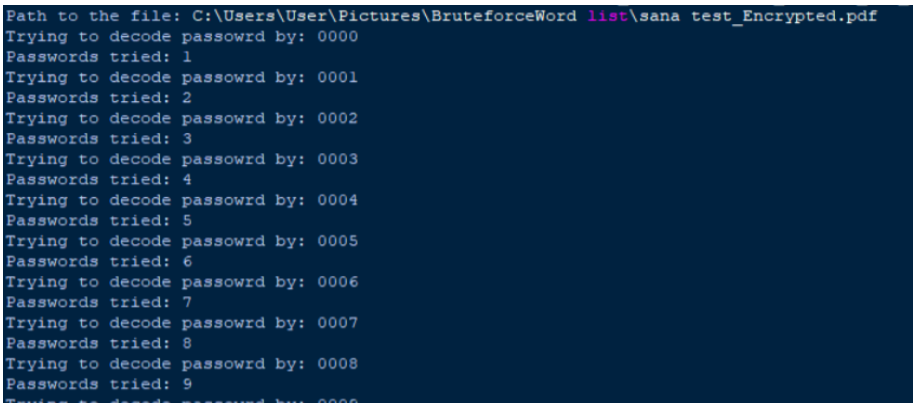
- Else its **password** protected then initialize the variable wordlist. File,open the file **“wordlist.txt”**, **”r”** represents read mode and ignore errors if any.
- Intilalize the **body** variable as a reference to **wordListFile content.lower()** function will convert and read all characters in lower case only.
- Use **split()** function to fetch different **passwords** on different lines and store it in an array with name words.

```
for i in range(len(words)):
    word = words[i]
    print('Trying to decode passowrd by: {}'.format(word))
    result = pdfReader.decrypt(word)
    if result == 1:
        print('Success! The password is: '+ word)
        break

    elif result == 0:
        tried += 1
        print('Passwords tried: ' + str(tried))
        continue
```

WhiteHat Jr + WhiteHat Jr

- Apply a for loop on words which will iterate till the length of the array **“words”**.
- Initialize the current value of array in variable **“word”**.
- **pdfReader.decrypt(word)** function will try to **decrypt** the password encrypted file using the current value of word as **password**.
- The result of the same will be stored in the variable **result**.
- If the function is able to decrypt the file successfully it will return the value of **result** as **1** and the break keyword will break the for loop,
- Else the value of **result** will be **0** and it will continue to the next word in array words.
- Print the value of the **result**.



5. While running the program needs to get a folder path. Right click on the folder whose password you want to decode, check the location, copy the location path followed by the folder name.
   Hence we cracked the password.

We have successfully learned to gain access to password protected files

What's NEXT?

In the next class we will be learning about phishing.

Expand Your Knowledge

To learn more about bruteforce attack click here

WhiteHat Jr + WhiteHat Jr + WhiteHat Jr