SINS CIPHERSEC - PASSWORD CHECKER AND PASSWORD GENERATOR

PROJECT REPORT

Done by

Mr. Suhas Dhole

Mr. Mohamed Ibrahim

Mr. Sabir

Mr. Naresh Mekala

Under the guidance of

Mrs. SINI S NAIR

(PRINCIPAL TECHNICAL OFFICER)

In partial fulfilment of the requirement for

CERTIFICATION COURSE

IN

CYBER SECURITY AND ETHICAL HACKING

NATIONAL INSTITUTE OF ELECTRONICS & INFORMATION TECHNOLOGY, CALICUT

(An Autonomous Scientific Society of Ministry of Electronics & Information Technology)

Government of India

Post box no.5, NIT Campus P.O., Calicut 673601, Kerala, India.

DECLARATION

We hereby declare that the project work entitled

SINS CIPHERSEC -

PASSWORD CHECKER AND PASSWORD GENERATOR

Done at

NATIONAL INSTITUTE OF ELECTRONICS &

INFORMATION TECHNOLOGY, CALICUT

(An Autonomous Scientific Society of Ministry of Electronics & Information Technology)

Government of India

Post box no.5, NIT Campus P.O., Calicut 673601, Kerala, India.

Under the guidance of

Mrs. SINI S NAIR

Name of the Students	**Signature**
Mr. Suhas Dhole	______
Mr. Mohamed Ibrahim	______
Mr. Sabir	______
Mr. Naresh Mekala	______

Place: CALICUT

Date: 15/01/2023

CERTIFICATE

This is to certify that

Mr. SUHAS DHOLE

Mr. MOHAMED IBRAHIM

Mr. SABIR

Mr. NARESH MEKALA

Of Certificate course on Cyber Security and Ethical Hacking have successfully completed their project work entitled ***SINS CIPHERSPEC - PASSWORD CHECKER AND PASSWORD GENERATOR*** at the ***Information Technology Lab, NIELIT Calicut***, under the guidance of ***Mrs. Sini S Nair, Principal Technical Officer***. The duration of the project is fifteen days.

Internal Guide
NIELIT Calicut

Course Coordinator
NIELIT Calicut

ACKNOWLEDGEMENT

First of all, we express our sincere gratitude to the almighty whose blessings helped us to materialise this project.

We would like to thank **Dr. M P Pillai**, Executive Director, NIELIT Calicut for providing all the facilities required.

Our sincere thanks to our project guide **Mrs. Sini S Nair**, Principal Technical Officer, NIELIT Calicut, for spending her precious time for us and giving her valuable ideas and suggestions, which helped completing our project successfully.

We express our deep gratitude to **Mr. Hari K** and **Mr. Mohamed Affan** for providing us with necessary technical support.

Finally we would like to express our sincere thanks to all of our friends, colleagues, parents and all those who have directly assisted during this project.

ABSTRACT

The SINS-CipherSec project is a comprehensive password management and security solution that aims to empower individuals and organizations to secure their online accounts. The project offers a user-friendly interface that allows users to easily generate strong and unique passwords using a passphrase and name as inputs. The generated passwords are then checked for strength against various criteria, including length, character types, and the presence of repeating characters, and are also cross-referenced against known compromised password databases.

In addition to password generation, the SINS-CipherSec project also provides a feature to save the generated passwords in a user-specific directory, with the current date as the file name, to allow easy tracking. This project offers both command-line interface (CLI) and graphical user interface (GUI) options, making it accessible to users of all technical backgrounds.

The SINS-CipherSec project is designed to make password management and security more accessible and user-friendly, providing a powerful tool to help individuals and organizations secure their online accounts and protect themselves from cyber threats.

INDEX

Contents

INTRODUCTION:

In today's digital age, online security is of paramount importance. With the increasing number of online accounts and services, the need for secure and unique passwords has never been greater. Unfortunately, many people struggle with creating strong passwords that are easy to remember and not easily guessable. Additionally, keeping track of multiple passwords can be difficult, and often people resort to using the same password for multiple accounts, which can lead to security breaches.

To address these issues, the SINS-CipherSec project aims to provide a tool to help people secure their online accounts. The tool includes features for generating strong, unique passwords and storing them in a secure manner, as well as a feature for assessing the strength of existing passwords and generating new, stronger passwords if necessary. The tool also provides both a CLI and GUI interface to make it easy for users to use.

Overall, the SINS-CipherSec project aims to provide users with a simple and easy-to-use solution for securing their online accounts and protecting themselves from potential security breaches.

PROJECT OVERVIEW

The SINS-CipherSec project is an open-source tool aimed at helping users secure their online accounts. The tool consists of four main scripts: "ciphersec.py", "pass-checker-CLI.py", "pass-checker-GUI.py", and "pass-gen-CLI.py" and "pass-gen-GUI.py".

The main script, "ciphersec.py", serves as the entry point for the tool. It displays an ASCII art logo and presents the user with a menu of options to choose from. The options include checking the strength of a password in both CLI and GUI mode, and generating new passwords in both CLI and GUI mode.

The "pass-checker-CLI.py" and "pass-checker-GUI.py" scripts are used to check the strength of a given password. The scripts check if the password is at least 8 characters long, contains at least one numeric character, one special character, one capital letter, and one small letter, and does not contain more than 2 consecutive repeating characters. It also checks if the password is in any of the text files in the directory "db-SecLists", which contain a list of breached passwords.

The "pass-gen-CLI.py" and "pass-gen-GUI.py" scripts are used to generate new, strong passwords. The scripts take the user's name and a passphrase as input, and use them to generate a new password using a simple encryption algorithm. The generated password is then saved to a text file with the current date as the file name in a directory named after the user's name under the directory "users". If a directory with the same name already exists, the previous password file is renamed as "old.password" and the new password is saved with current date.

In summary, the SINS-CipherSec project provides an easy-to-use, open-source tool for users to secure their online accounts by generating strong, unique passwords and checking the strength of their existing passwords, as well as providing an easy way to store these password in a secure and organized anner.

TECHNICAL DESCRIPTION

The SINS-CipherSec tool is a password management and generation tool designed to help users secure their online accounts. The tool is composed of four main scripts:

- "ciphersec.py" is the main script that runs the program and presents the user with the options to check their password strength, generate a new password, or exit the program.
- "pass-checker-CLI.py" is the script for checking password strength in command-line interface (CLI) mode. It takes a user-provided password as input and checks it against a set of predefined rules, such as length, the presence of special characters, etc. It also checks if the password is present in a compromised password database.
- "pass-checker-GUI.py" is the script for checking password strength in a graphical user interface (GUI) mode. It uses the Tkinter library to create a simple GUI for entering a password and returning the strength assessment.
- "pass-gen-CLI.py" and "pass-gen-GUI.py" scripts are used for generating new passwords. The CLI script prompts the user for their name and a passphrase, and uses these inputs to generate a new password using a simple algorithm. The GUI script uses the Tkinter library to create a GUI for entering the name and passphrase, and displays the generated password.

The tool also includes a "db-SecLists" directory which contains the compromised password databases and a "logo" directory that contains ASCII art logo that is displayed at the start of the program. The "users" directory is used to store the generated passwords for each user with date.

The tool is designed to be user-friendly and easy to use, with clear instructions and error messages to guide the user through the process of checking their password strength or generating a new password. It is also designed to be secure, with the password databases and generated passwords stored in a secure manner.

- **Password Generation:**

  In the SINS-CipherSec project, there are two methods of password generation provided: one through a command-line interface (CLI) and the other through a graphical user interface (GUI). The CLI password generation script, "pass-gen-CLI.py", prompts the user to enter their name and a passphrase. The script then uses the user's name and passphrase to generate a new password. The generated password is a combination of letters and numbers that are derived from the user's name and passphrase. The password is generated by adding the ASCII values of characters from the name and passphrase and taking the modulus of the sum with 26. The result is then added to the ASCII value of 'A' to get the final character of the password.

  The GUI password generation script, "pass-gen-GUI.py", also prompts the user to enter their name and a passphrase. The script then uses the user's name and passphrase to generate a new password in the same way as the CLI script. The generated password is displayed on the GUI and the user has the option to save it to a file or copy it to the clipboard. The password is saved to a directory named "users" with a subdirectory named after the user and the date is added to the file name. If a file with the same name already exists, it will be renamed as "old.password" with the current date.

- **Password Strength Checker:**

  The SINS-CipherSec tool includes a feature that allows users to check the strength of their passwords. This feature is implemented in two different modes, a Command Line Interface (CLI) mode and a Graphical User Interface (GUI) mode.

  The CLI mode, implemented in the "pass-checker-CLI.py" script, prompts the user to enter a password and then runs a series of checks on the password to determine its strength. The script checks for the following:

  - The password is at least 8 characters long
  - The password contains at least one numeric character
  - The password contains at least one special character
  - The password contains no more than 2 consecutive repeating characters
  - The password is not found in any of the .txt files in the "db-SecLists" directory, which contain a list of known breached passwords.
  - If the password passes all of these checks, the script returns "Secure password." Otherwise, it returns a message specifying which check the password failed.

  The GUI mode, implemented in the "pass-checker-GUI.py" script, provides a graphical interface for the user to enter a password and check its strength. It performs the same checks as the CLI mode and displays the result in a message box.

  Both the CLI and GUI mode helps users in assessing their password strength and take action accordingly, by either changing the password or generating a new one using the password generation feature of SINS-CipherSec.

- **Password Saving and Tracking:**

  In the SINS-CipherSec project, the generated passwords are saved and tracked in a secure manner. The project includes a feature to save the generated passwords in a directory named "users" and subdirectory with the name entered by the user. Each time a password is generated, the current date is added to the file name, and if there is an existing password file, it is renamed to "old.password" with the current date. This allows for easy tracking of previous passwords and helps to ensure that the user is not reusing the same password across multiple accounts. Additionally, the password files are stored on the user's local machine rather than on a remote server, providing an additional layer of security.

- **CLI and GUI Interface**

  In the SINS-CipherSec project, both command line interface (CLI) and graphical user interface (GUI) options are provided for the users to interact with the tool. The CLI option allows users to interact with the tool using commands and the GUI option provides a graphical interface for the users to interact with the tool. The CLI option is suitable for users who prefer to use keyboard commands, while the GUI option is suitable for users who prefer a more visual approach. Both options provide the same functionality and the choice of interface is left to the user's preference. The GUI version is built using the tkinter library in python which provides a simple and easy way to create a graphical interface. The CLI version is built using the subprocess library in python which allows to run other python script in command line interface.

APPENDICES

System Requirements:

- A computer running a Windows, MacOS, or Linux operating system
- Python 3.6 or later
- Tkinter library for the GUI version
- Access to the internet for checking password against breached password databases
- At least 50 MB of available storage space for storing generated passwords and tracking password history
- A text editor or word processor for creating the project report.

Important Directories and Files:

- *Dir: Documentation*
- *Dir: db-SecLists*
    - *rockyou.txt*
    - *2020-200_most_used_passwords.txt*
    - *500-worst-passwords.txt*
    - *500-worst-passwords.txt.bz2*
    - *Keyboard-Combinations.txt*
    - *Most-Popular-Letter-Passes.txt*
    - *PHP-Magic-Hashes.txt*
    - *SCRABBLE-hackerhouse.tgz*
    - *UserPassCombo-Jay.txt*
    - *bt4-password.txt*
    - *cirt-default-passwords.txt*
- *Dir: logo*
    - *eagl-sins.txt*
    - *linux0-sins.txt*
    - *linux1 -sins.txt*
    - *linux2-sins.txt*
    - *succubus-sins.txt*
    - *tech-sins.txt*

- *Dir: script-main*
    - *pass-checker-CLI.py*
    - *pass-checker-GUI.py*
    - *pass-gen-CL1.py*
    - *pass-gen-GU1.py*
- *Dir: static*
    - *logo_for_sins_with_white_Minimalism.png*
    - *sins-main-logo.png*
    - *sins-main2-logo.jpg*
- *File: README.md*
- *File: ciphersec.py*
- *File: requirements.txt*

Code:

1. ciphersec.py

The script "ciphersec.py" is the core module of our SINS-CipherSec project. It contains the functions and logic for generating strong, unique passwords using a passphrase and user's name as input. The generated password is then checked for strength using various checks like length, special characters, numeric characters and more. If the password passes all the checks, it is considered a strong password and can be saved for future use. The script also contains functions for saving and tracking passwords, as well as a CLI and GUI interface for user interaction.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import os
import random
import glob
import subprocess
from termcolor import colored

def main():

# Code for show ascii art logo from /logo directory
    # Get the directory path of the main.py file
    script_dir = os.path.dirname(os.path.abspath(__file__))
    # Join the directory name to form the relative path to the
logo directory
    directory = os.path.join(script_dir, 'logo')
    # Get the list of all the .txt files in the logo directory
    txt_files = glob.glob(os.path.join(directory, '*.txt'))
    # Choose a random file from the list
    chosen_file = random.choice(txt_files)
    # Open the file
    with open(chosen_file, 'r') as file:
        # Read the contents of the file
        contents = file.read()
        # Define a list of colors
        colors = ['red', 'green', 'yellow', 'blue', 'magenta',
'cyan']
        # Choose a random color
        color = random.choice(colors)
        # Print the contents of the file in the chosen color
        print(colored(contents, color))


# Code for runing scripts from /script-main directory
    # Get the directory path of the main.py file
    script_dir = os.path.dirname(os.path.abspath(__file__))
    # Join the directory name to form the relative path to the
desired directory
    directory = os.path.join(script_dir, 'script-main')

    print("What do you want to do next?")
    print("1. (CLI) Check ur Passw0rd! strength")
    print("2. (GUI) Check ur Passw0rd! strength")
    print("3. (CLI) Generate the unbreakable Passw0rd!")
    print("4. (GUI) Generate the unbreakable Passw0rd!")
    choice = input("Enter your choice(1/2/3/4): ")

    if choice == "1":
        subprocess.call(["python",      os.path.join(directory,
"pass-checker-CLI.py")])
    elif choice == "2":
        subprocess.call(["python",      os.path.join(directory,
"pass-checker-GUI.py")])
    elif choice == "3":
        subprocess.call(["python",      os.path.join(directory,
"pass-gen-CLI.py")])
    elif choice == "4":
        subprocess.call(["python",      os.path.join(directory,
"pass-gen-GUI.py")])
    else:
        print("Invalid choice")

if __name__ == '__main__':
    main()
```

2. **pass-checker-CLI.py**

The script "pass-checker-CLI.py" is a command-line interface tool for checking the strength of a given password. The script accepts user input of a password and checks it against several conditions, including minimum length, inclusion of at least one numeric character, inclusion of at least one special character, and the presence of more than 2 consecutive repeating characters. The script also checks if the password is present in any of the .txt files in the directory "db-SecLists" as a means of checking for a breach of the password. If the password passes all the conditions, the script returns "Secure password." If the password fails any of the conditions, the script returns the specific condition the password failed.

```
# script for password checker CLI mode

import os
import re

def check_password(password):

    # check if the password is at least 8 characters long
    if len(password) < 8:
        return "Password must be at least 8 characters long."

    # check if the password contains at least one numeric
character
    elif not any(i.isdigit() for i in password):
        return "Password must contain at least one numeric
character."

    # check if the password contains at least one special
character
    elif not any(i in "!@#$%^&*()_+-=[]{};:'\"\\|,.<>/?" for i
in password):
        return "Password must contain at least one special
character."

    # check if the password contains more than 2 consecutive
repeating characters
    elif re.search(r"(\w)\1{2,}", password):
        return "Password can contain only 2 consecutive
repeating characters."
```

```
    # check if the password is in any of the .txt files in the
directory
    directory                                                  =
os.path.join(os.path.dirname(os.path.abspath(__file__)), '..',
'db-SecLists')
    for filename in os.listdir(directory):
        if filename.endswith(".txt"):
            with  open(os.path.join(directory,  filename))  as
file:
                if password in file.read():
                    return "Breached Password. The Password you
entered is in the compromised database. Please use difficult
another password"

    # if the password pass all the above conditions, then return
valid
    else:
        return "Secure password."

password = input("Enter the password: ")
result = check_password(password)
print(result)
```

3. pass-checker-GUI.py

The script "pass-checker-GUI.py" is a graphical user interface (GUI) version of the password checker tool. It uses the Tkinter library to create a simple window with an input field for the user to enter a password, a submit button to check the password, and a message box to display the result. The check_password() function, which is used in both the CLI and GUI versions of the tool, checks the entered password against several conditions, including length, the presence of special characters and numbers, and the presence of repeating characters. If the password passes all the conditions, it is considered a "Secure password." Otherwise, an appropriate error message is displayed. Additionally, the script also checks if the password is in any of the .txt files in the directory, if the entered password is already compromised, script will return the breach message to use a difficult password.

```
# script for password checker GUI mode

import tkinter as tk
from tkinter import messagebox
import re
import os

def check_password(password):

    # check if the password is at least 8 characters long
    if len(password) < 8:
        return "Password must be at least 8 characters long."

    # check if the password contains at least one numeric
character
    elif not any(i.isdigit() for i in password):
        return "Password must contain at least one numeric
character."

    # check if the password contains at least one special
character
    elif not any(i in "!@#$%^&*()_+-=[]{};:'\"\\|,.<>/?" for i
in password):
        return "Password must contain at least one special
character."
```

```
    # check if the password contains more than 2 consecutive
repeating characters
    elif re.search(r"(\w)\1{2,}", password):
        return "Password can contain only 2 consecutive
repeating characters."

    # check if the password contains at least one capital letter
    elif not any(i.isupper() for i in password):
        return "Password must contain at least one capital
letter."

    # check if the password contains at least one small letter
    elif not any(i.islower() for i in password):
        return "Password must contain at least one small
letter."

    # check if the password is in any of the .txt files in the
directory
    directory =
os.path.join(os.path.dirname(os.path.abspath(__file__)), '..',
'db-SecLists')
    for filename in os.listdir(directory):
        if filename.endswith(".txt"):
            with open(os.path.join(directory, filename)) as
file:
                if password in file.read():
                    return "Breached Password. The Password you
entered is in the compromised database. Please use difficult
another password"

    # if the password pass all the above conditions, then return
valid
    else:
        return "Secure password."


def on_submit():
    password = entry.get()
    result = check_password(password)
    messagebox.showinfo("Password Checker", result)

root = tk.Tk()
root.geometry("300x400")
root.title("SINS | Password Checker")
```

```
label = tk.Label(root, text="Enter password:")
label.pack()

entry = tk.Entry(root)
entry.pack()

submit = tk.Button(root, text="Submit", command=on_submit)
submit.pack()

root.mainloop()
```

4. pass-gen-CLl.py

The script "pass-gen-CLI.py" is a command line interface tool for generating passwords. It prompts the user to enter their name and a passphrase, and uses these inputs to generate a new password. The script then creates a directory with the user's name if it doesn't exist, and saves the generated password in a file with the current date as the file name. The script also renames any previous password files as "old.password" before saving the new password. The generated password is also displayed on the command line for the user to see.

```
# Script for generating password CLI mode

import os
import datetime

# function to generate password
def generate_password():
    name = input("Enter Your Name: ")
    passphrase = input("Enter Your Passphrase: ")
    password = ""
    j = 0
    for i in range(len(passphrase)):
        char = passphrase[i]
        if j == len(name):
            j = 0
        x = (ord(char) + ord(name[j])) % 26
        x += ord('A')
        password += chr(x)
        j += 1
    print("Generated Password: ", password)
    return password,name

# function to save password
def save_password(password,name):
    #get the current date
    now = datetime.datetime.now()
    current_date = now.strftime("%Y-%m-%d")
    # create the directory "users" if it does not exist
    if not os.path.exists("users"):
        os.makedirs("users")
    # create a directory with the user's name if it does not
exist
```

```
    user_directory = "users/" + name
    if not os.path.exists(user_directory):
        os.makedirs(user_directory)
    # rename the previous password file to "old.password" if it
exists
    if os.path.exists(user_directory+'/password.txt'):
        os.rename(user_directory+'/password.txt',
user_directory+'/old.password')
    # save the new password to a file with the current date as
the file name
    file            =            open(user_directory            +
"/password"+current_date+".txt", "w")
    file.write(password)
    file.close()
    print("Password                  saved                    to
"+user_directory+"/password"+current_date+".txt.")

# main function to call other functions
def main():
    password,name = generate_password()
    save_password(password,name)

# call the main function
if __name__ == '__main__':
    main()
```

5. **pass-gen-GUl.py**

The script "pass-gen-GUI.py" is a graphical user interface (GUI) based script that generates a password based on the user's name and passphrase. The script uses the tkinter library to create a simple GUI for user input of name and passphrase. The generated password is displayed on the screen and can also be saved to a file or copied to the clipboard. The script also handles renaming of previous password files and includes a timestamp in the file name.

```
# sctipt for password generator GUI mode

import tkinter as tk
from tkinter import messagebox
import os
import datetime

def generate_password():
    name = name_entry.get()
    passphrase = passphrase_entry.get()
    password = ""
    j = 0
    for i in range(len(passphrase)):
        char = passphrase[i]
        if j == len(name):
            j = 0
        x = (ord(char) + ord(name[j])) % 26
        x += ord('A')
        password += chr(x)
        j += 1
    password_label.config(text=password)

def save_password():
    password = password_label.cget("text")
    name = name_entry.get()
    # Create a directory named "users" if it doesn't exist
    if not os.path.exists("users"):
        os.mkdir("users")
    # Create a subdirectory with the name entered by the user
if it doesn't exist
    user_directory = "users/" + name
    if not os.path.exists(user_directory):
        os.mkdir(user_directory)
    # get the current date
    now = datetime.datetime.now()
    current_date = now.strftime("%Y-%m-%d")
    # rename the old password file as old.password with date
    for file in os.listdir(user_directory):
        if file.endswith(".password"):
            os.rename(user_directory + '/' + file,
user_directory + '/old.' + file + '.' + current_date)
    # save the new generated password with current date
    file = open(user_directory + "/password." + current_date +
".password", "w")
    file.write(password)
    file.close()
    messagebox.showinfo("Success", "Password saved to " +
user_directory + ".")

def copy_password():
    password = password_label.cget("text")
    root.clipboard_clear()
    root.clipboard_append(password)
    messagebox.showinfo("Success", "Password copied to
clipboard.")


root = tk.Tk()
root.geometry("200x200")
root.title("Password Generator")

name_label = tk.Label(root, text="Enter Your Name:")
name_label.pack()

name_entry = tk.Entry(root)
name_entry.pack()

passphrase_label = tk.Label(root, text="Enter Your
Passphrase:")
passphrase_label.pack()

passphrase_entry = tk.Entry(root)
passphrase_entry.pack()

generate_button = tk.Button(root, text="Generate Password",
command=generate_password)
generate_button.pack()
```

```
password_label = tk.Label(root)
password_label.pack()

save_button    =    tk.Button(root,    text="Save    Password",
command=save_password)
save_button.pack()

copy_button    =    tk.Button(root,    text="Copy    Password",
command=copy_password)
copy_button.pack()

root.mainloop()
```

Installation and Setup:

1. Clone the repository from GitHub by running the command "`git clone https://github.com/GTekSD/SINS-CipherSec.git`" in the command prompt.

2. Navigate to the SINS-CipherSec directory using the command "cd SINS-CipherSec".

3. Install the required libraries by running the command "`pip install -r requirements.txt`"

4. Make the script executable by running the following command "`chmod +x ciphersec.py`"

5. Add the path of your script to the system's PATH environment variable.

6. You can add the directory to the PATH in the shell startup file, such as `.bashrc` or `.bash_profile` for bash shell, or `.zshrc` for `zsh shell`.

7. Manually: add in `.zshrc`
```
# SINS CipherSec
alias ciphersec='python3 /home/kali/Tools/SINS-
CipherSec/ciphersec.py'
export PATH=$PATH:/home/kali/Tools/SINS-CipherSec
```

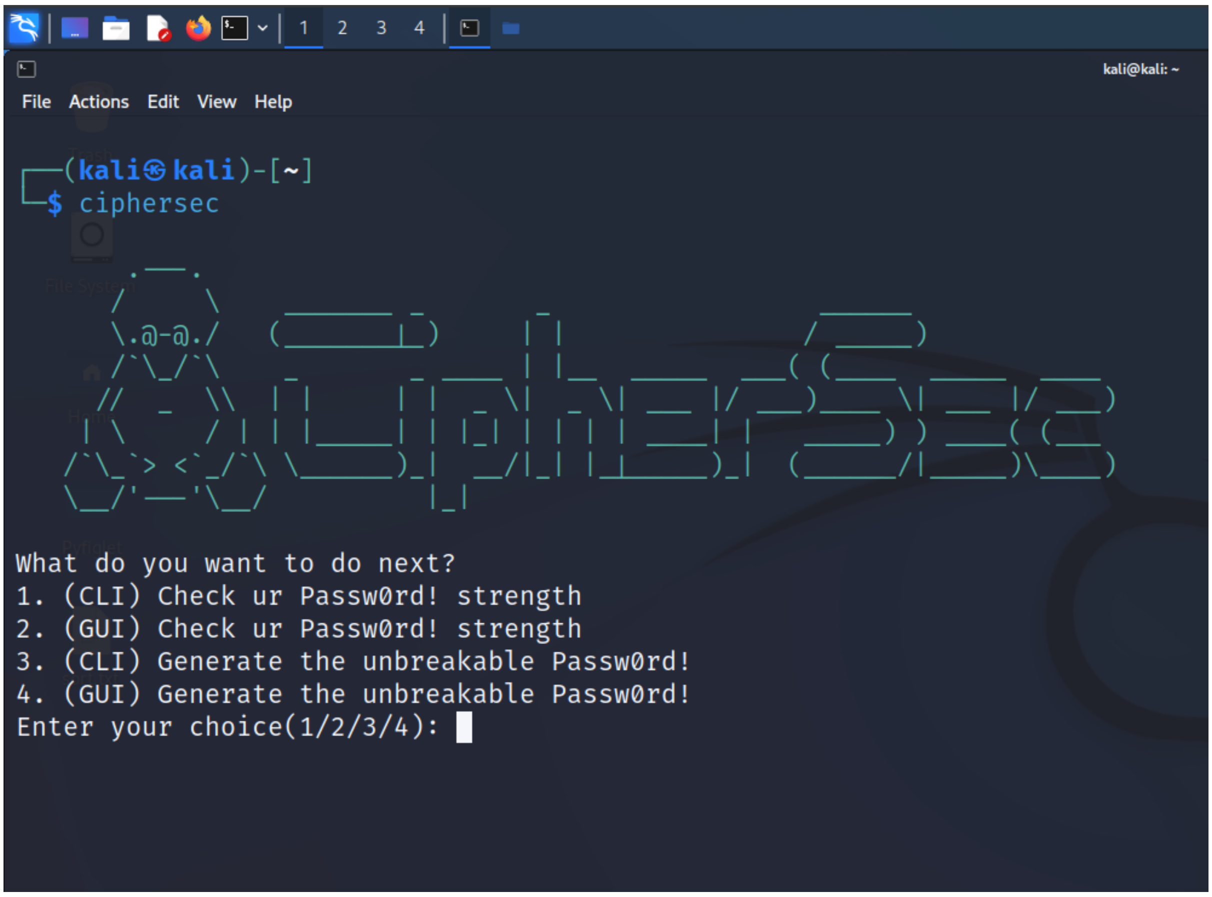
8. Follow the prompts on the command line interface or the graphical user interface to generate, check or save passwords.

9. Run the program by executing " `ciphersec.py`" or cmd " `ciphersec`"

10. Note: It is recommended to use python3 and above version to run the program.

Screenshots:

The program by executing " `ciphersec.py`" or cmd " `ciphersec`"

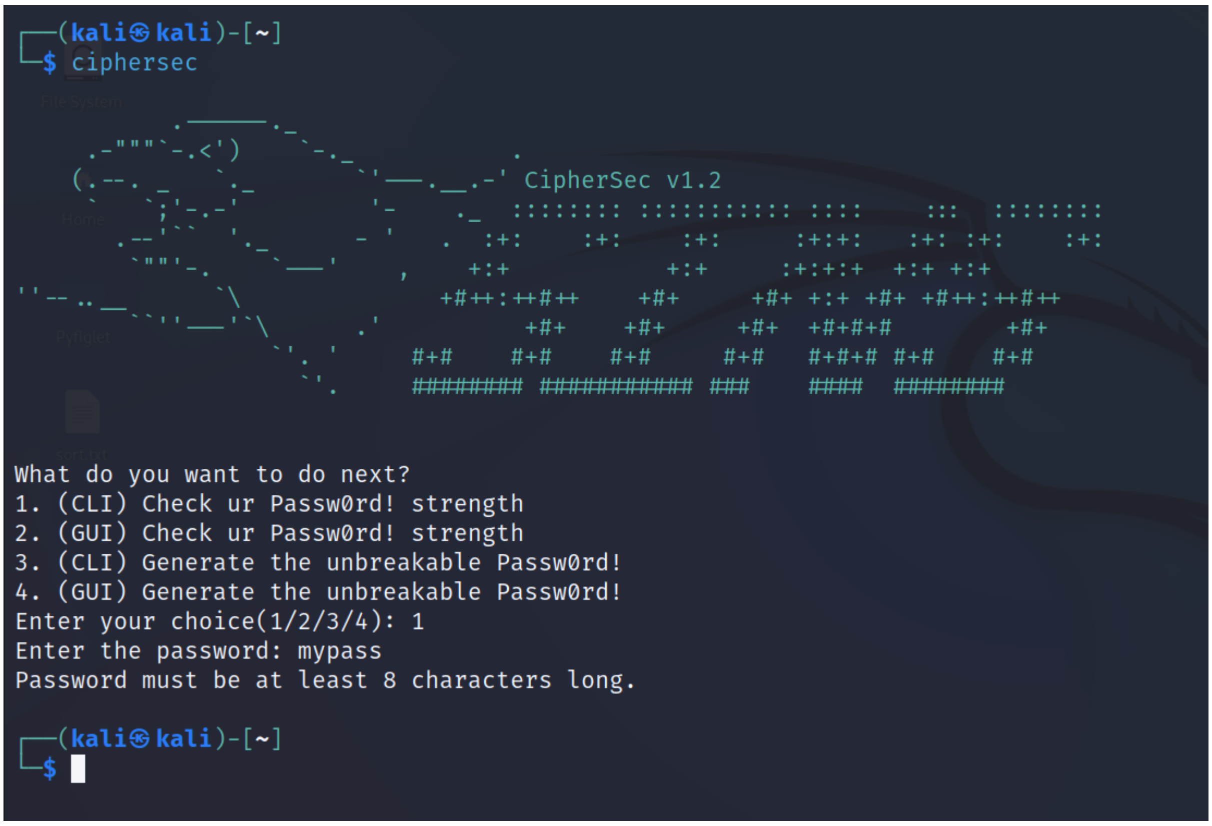


```
┌──(kali㉿kali)-[~]
└─$ ciphersec

What do you want to do next?
1. (CLI) Check ur Passw0rd! strength
2. (GUI) Check ur Passw0rd! strength
3. (CLI) Generate the unbreakable Passw0rd!
4. (GUI) Generate the unbreakable Passw0rd!
Enter your choice(1/2/3/4): 
```

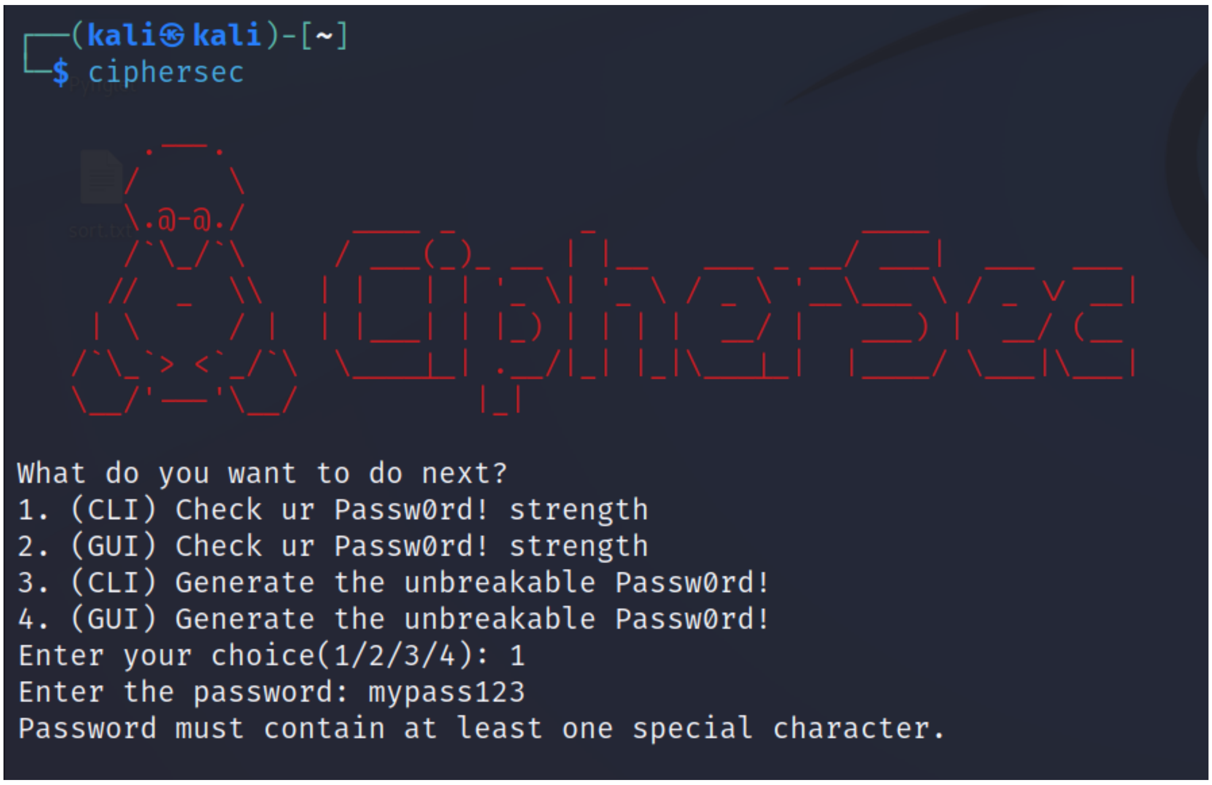
Password Checker CLI:

User must follow:

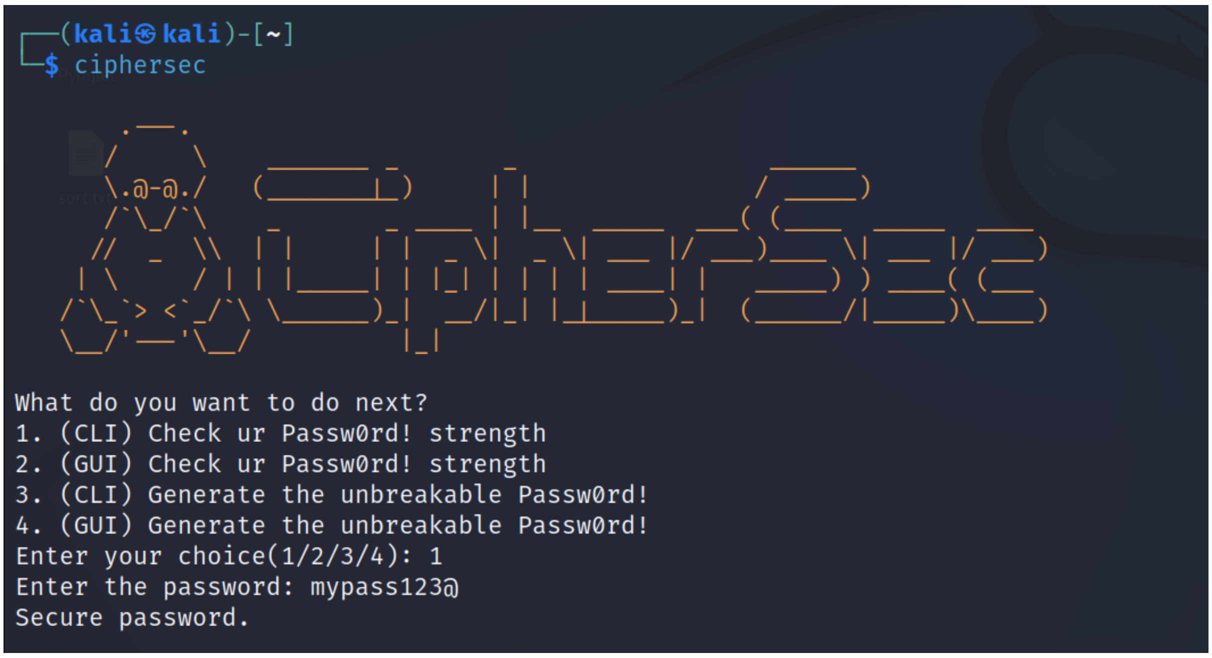
- Password must be at least 8 characters long.
- Password must contain at least one numeric character.
- Password must contain at least one special character.
- Password can contain only 2 consecutive repeating characters.
- Password must contain at least one capital letter.
- Password must contain at least one small letter.



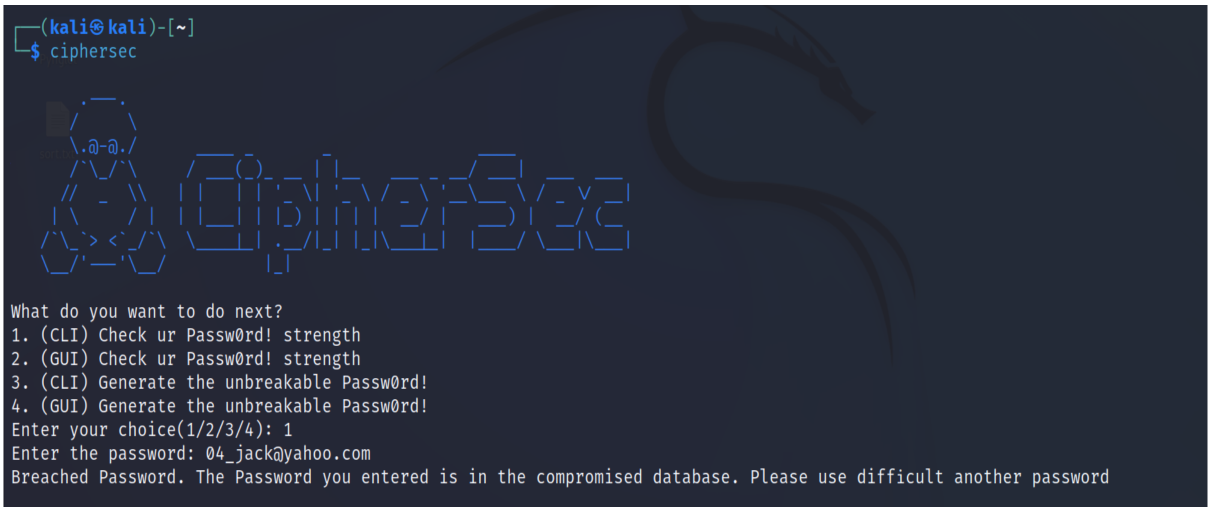
```
┌──(kali㉿kali)-[~]
└─$ ciphersec

                                          CipherSec v1.2

What do you want to do next?
1. (CLI) Check ur Passw0rd! strength
2. (GUI) Check ur Passw0rd! strength
3. (CLI) Generate the unbreakable Passw0rd!
4. (GUI) Generate the unbreakable Passw0rd!
Enter your choice(1/2/3/4): 1
Enter the password: mypass
Password must be at least 8 characters long.

┌──(kali㉿kali)-[~]
└─$ 
```



```
┌──(kali㉿kali)-[~]
└─$ ciphersec

What do you want to do next?
1. (CLI) Check ur Passw0rd! strength
2. (GUI) Check ur Passw0rd! strength
3. (CLI) Generate the unbreakable Passw0rd!
4. (GUI) Generate the unbreakable Passw0rd!
Enter your choice(1/2/3/4): 1
Enter the password: mypass123
Password must contain at least one special character.
```

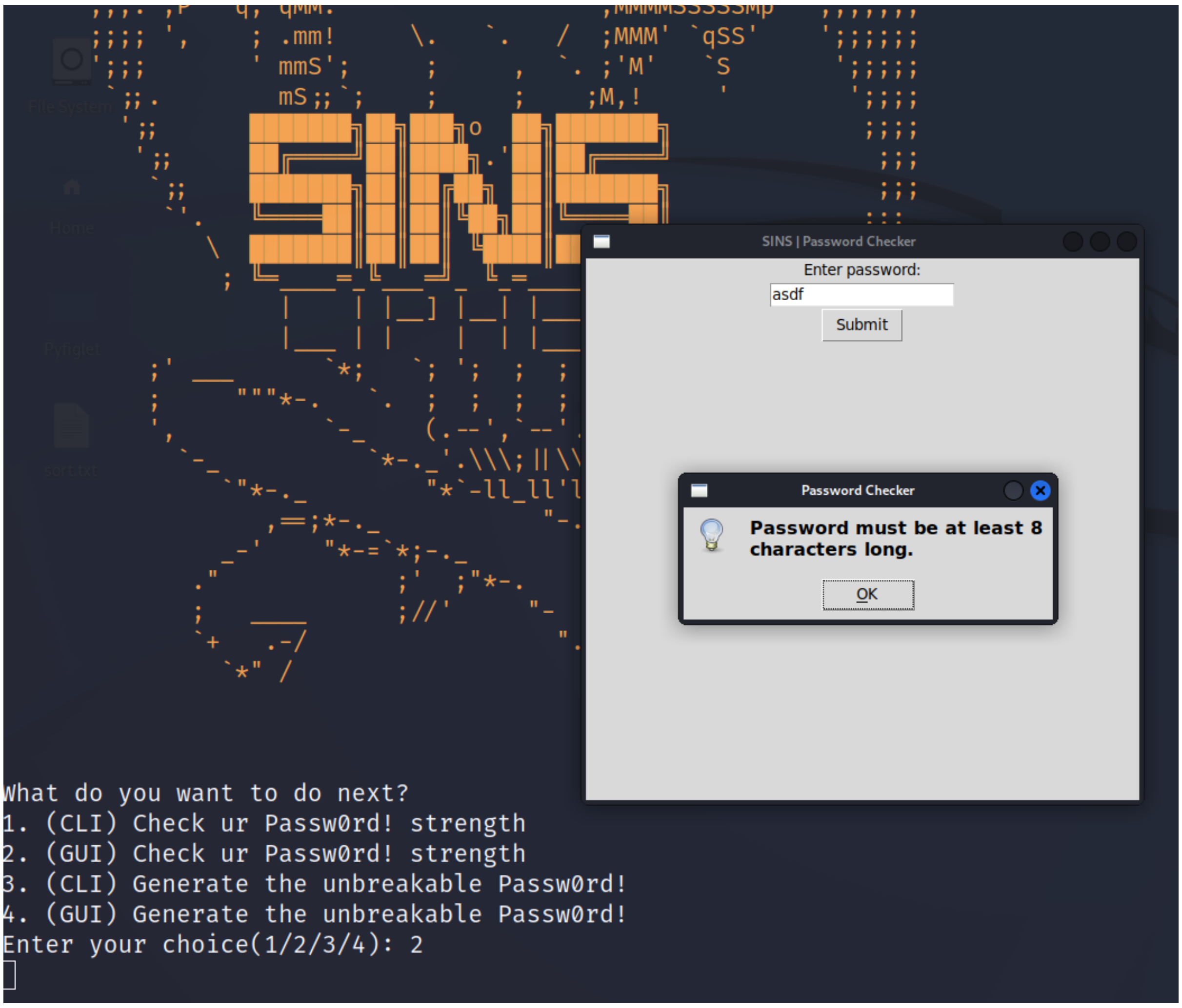


```
┌──(kali㉿kali)-[~]
└─$ ciphersec

What do you want to do next?
1. (CLI) Check ur Passw0rd! strength
2. (GUI) Check ur Passw0rd! strength
3. (CLI) Generate the unbreakable Passw0rd!
4. (GUI) Generate the unbreakable Passw0rd!
Enter your choice(1/2/3/4): 1
Enter the password: mypass123@
Secure password.
```



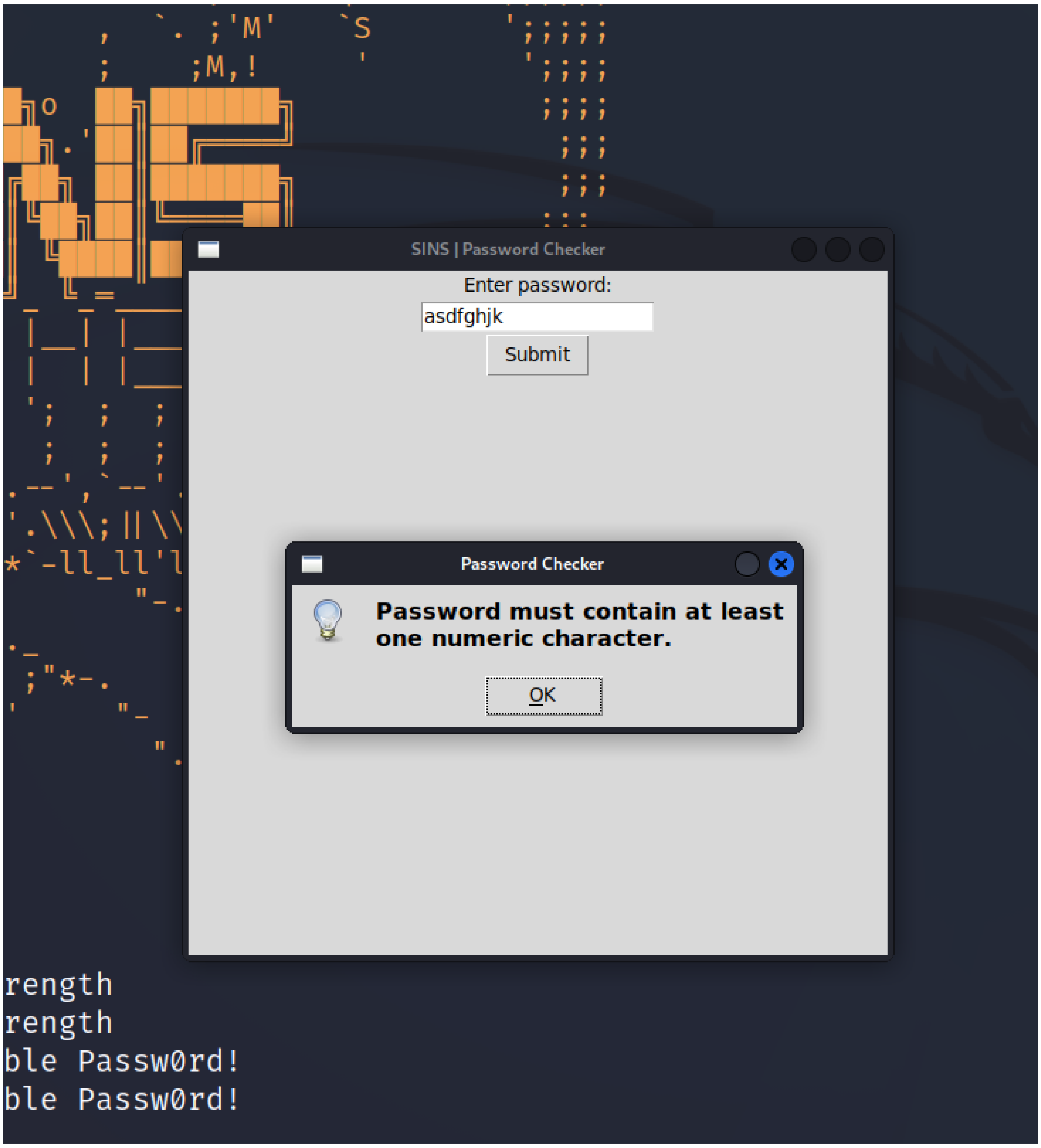
```
┌──(kali㉿kali)-[~]
└─$ ciphersec

What do you want to do next?
1. (CLI) Check ur Passw0rd! strength
2. (GUI) Check ur Passw0rd! strength
3. (CLI) Generate the unbreakable Passw0rd!
4. (GUI) Generate the unbreakable Passw0rd!
Enter your choice(1/2/3/4): 1
Enter the password: 04_jack@yahoo.com
Breached Password. The Password you entered is in the compromised database. Please use difficult another password
```

Password Checker GUI:



Password must be at least 8 characters long.

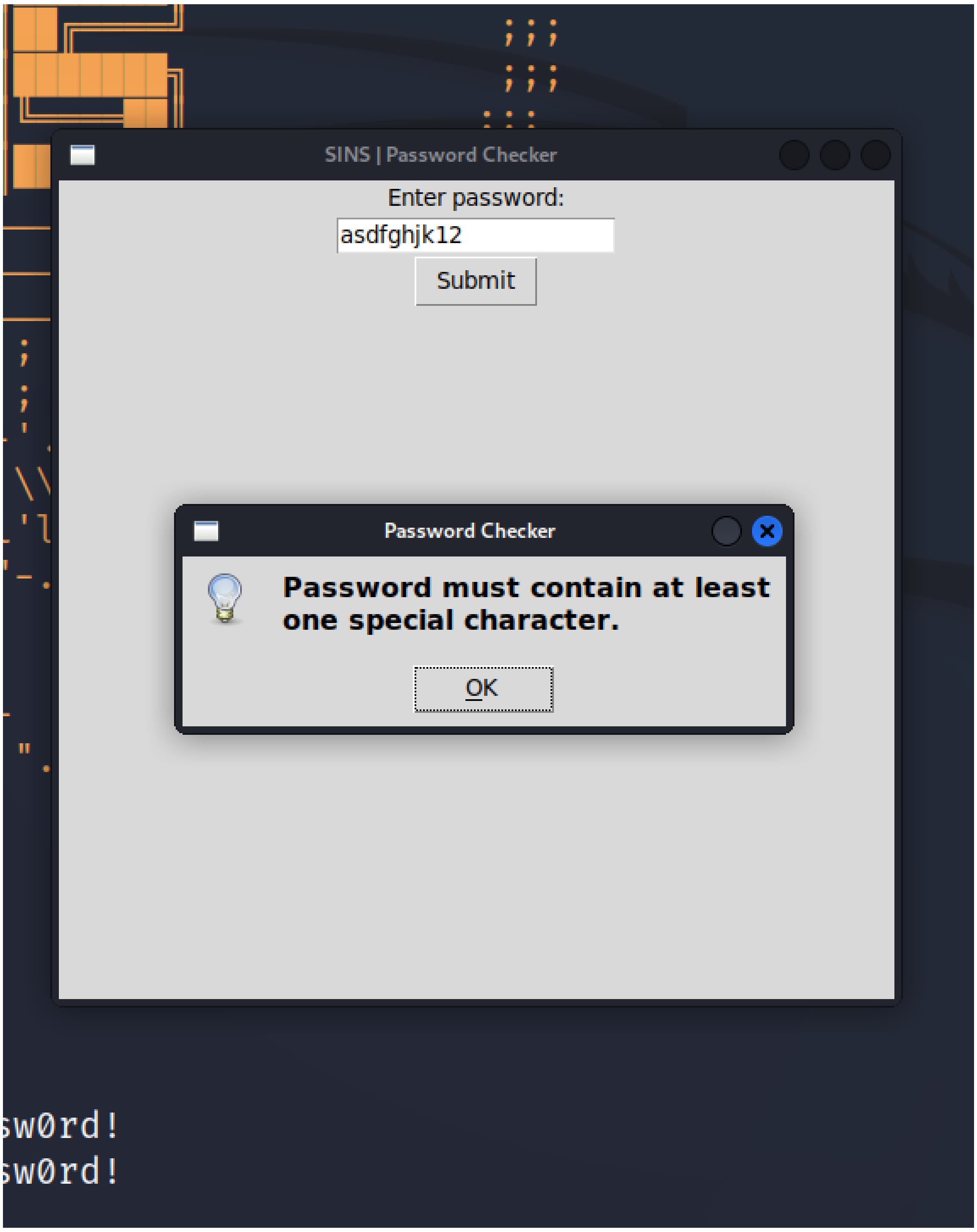
```
What do you want to do next?
1. (CLI) Check ur Passw0rd! strength
2. (GUI) Check ur Passw0rd! strength
3. (CLI) Generate the unbreakable Passw0rd!
4. (GUI) Generate the unbreakable Passw0rd!
Enter your choice(1/2/3/4): 2
```



asdfghjk

Password must contain at least one numeric character.

```
rength
rength
ble Passw0rd!
ble Passw0rd!
```


SINS | Password Checker

Enter password:

asdfghjk12

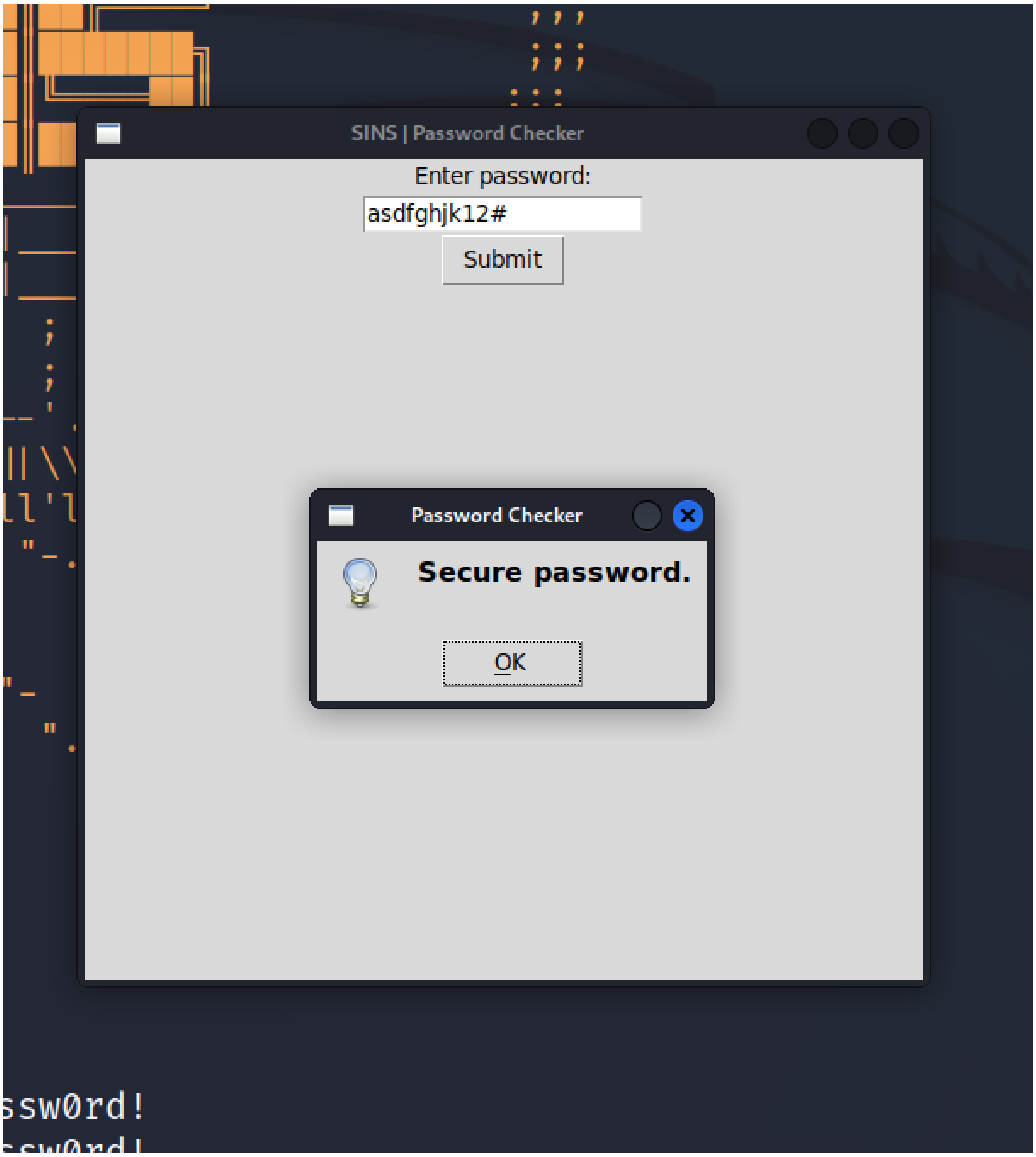
Submit

Password Checker

Password must contain at least one special character.

OK

```
sw0rd!
sw0rd!
```



SINS | Password Checker

Enter password:

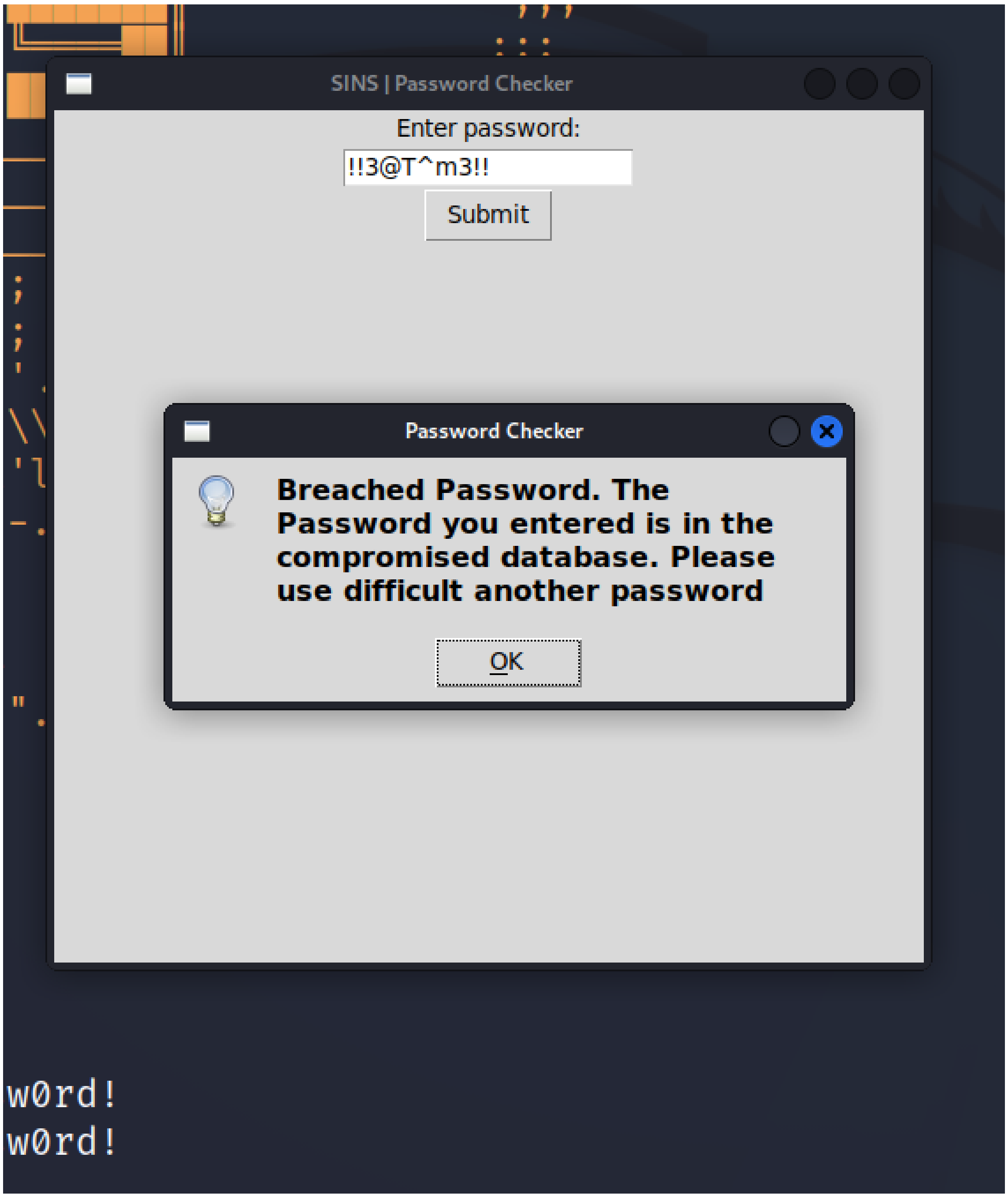
asdfghjk12#

Submit

Password Checker

Secure password.

OK



SINS | Password Checker

Enter password:

!!3@T^m3!!

Submit

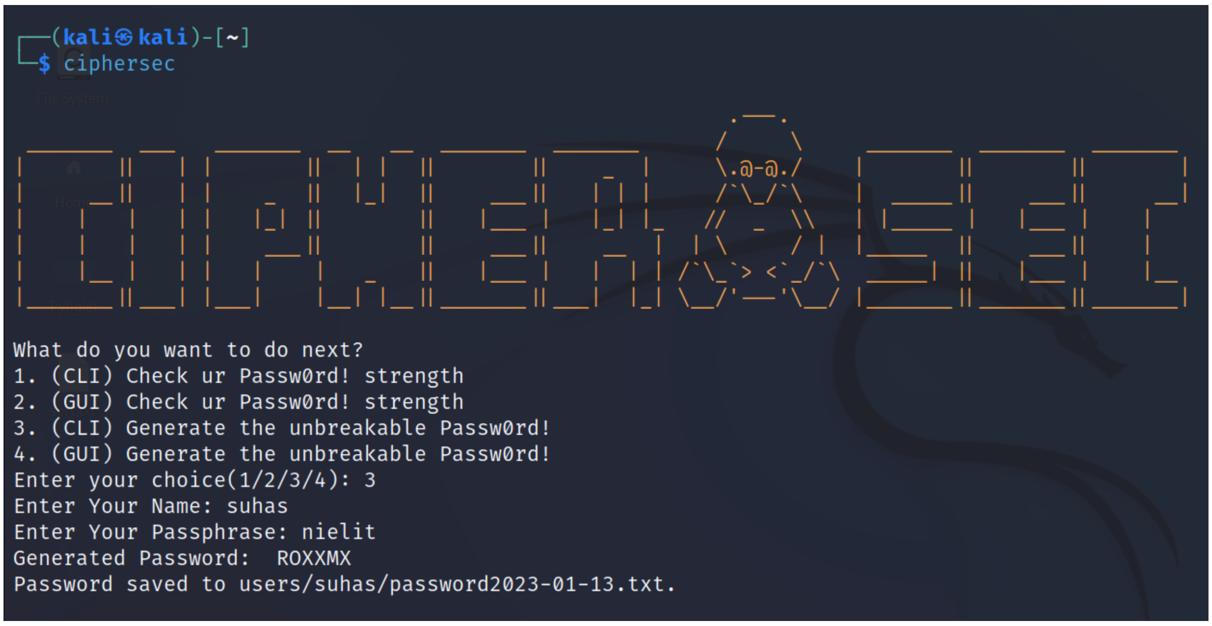
Password Checker

Breached Password. The Password you entered is in the compromised database. Please use difficult another password

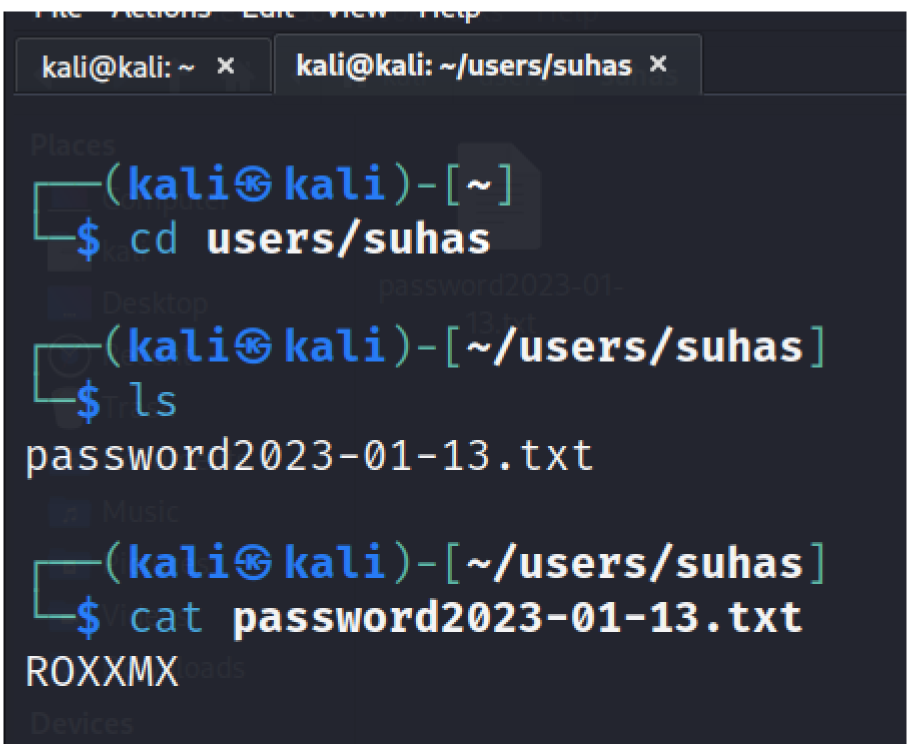
OK

```
w0rd!
w0rd!
```

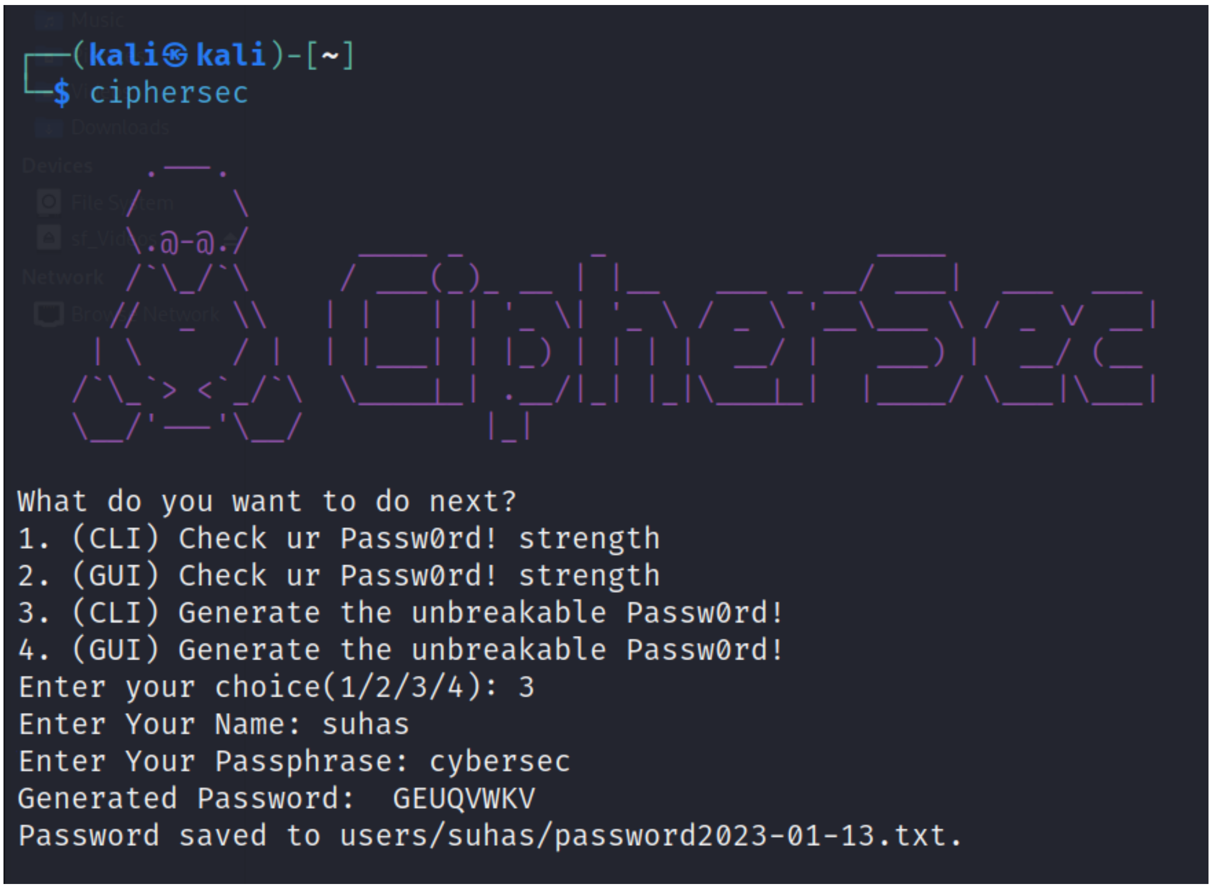
Password Generator CLI



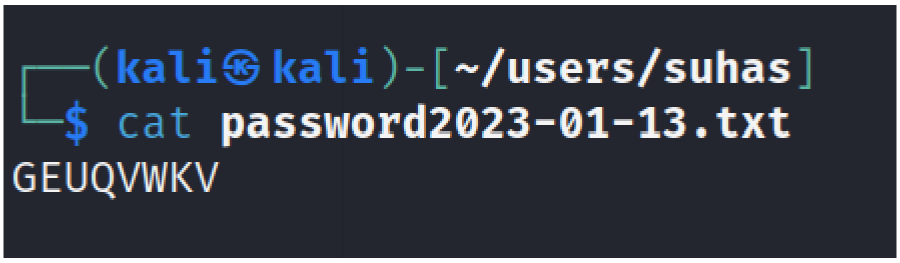
```
┌──(kali㉿kali)-[~]
└─$ ciphersec

What do you want to do next?
1. (CLI) Check ur Passw0rd! strength
2. (GUI) Check ur Passw0rd! strength
3. (CLI) Generate the unbreakable Passw0rd!
4. (GUI) Generate the unbreakable Passw0rd!
Enter your choice(1/2/3/4): 3
Enter Your Name: suhas
Enter Your Passphrase: nielit
Generated Password:  ROXXMX
Password saved to users/suhas/password2023-01-13.txt.
```



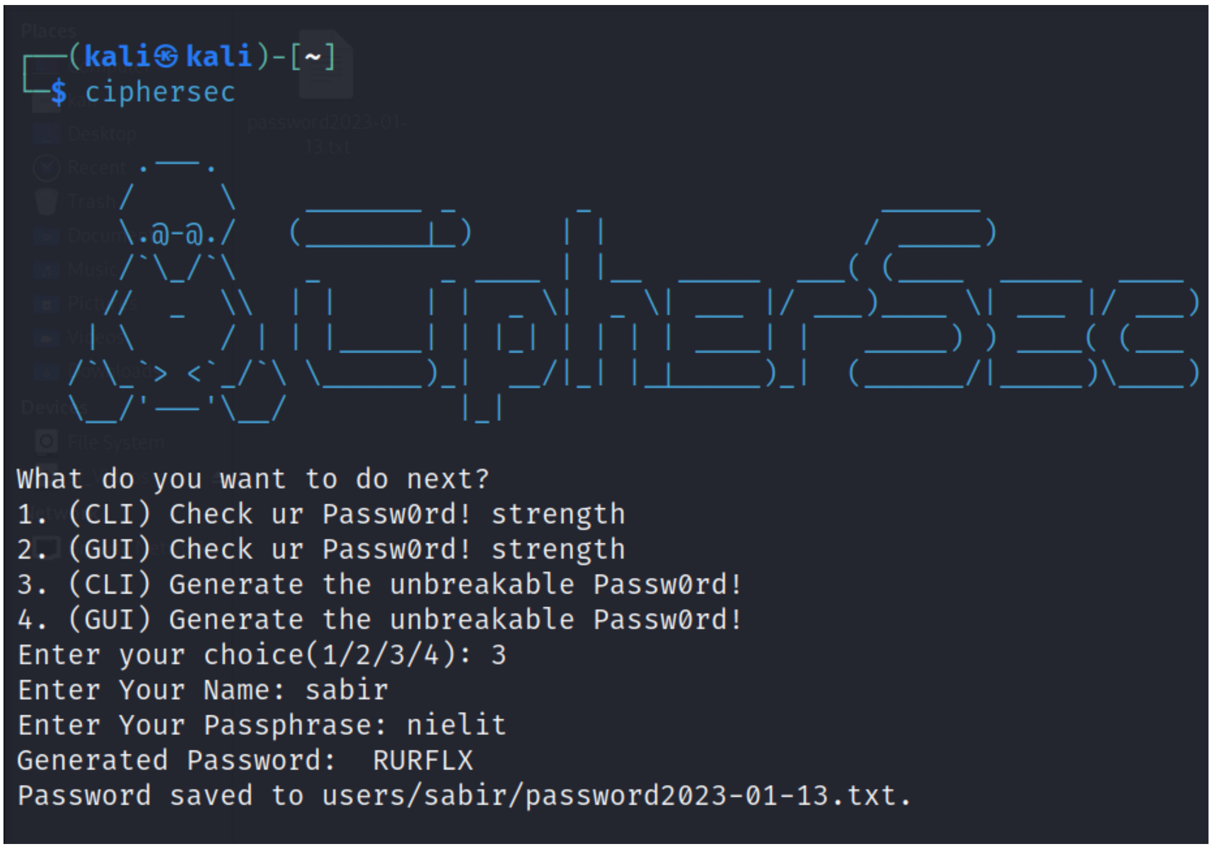
```
┌──(kali㉿kali)-[~]
└─$ cd users/suhas

┌──(kali㉿kali)-[~/users/suhas]
└─$ ls
password2023-01-13.txt

┌──(kali㉿kali)-[~/users/suhas]
└─$ cat password2023-01-13.txt
ROXXMX
```



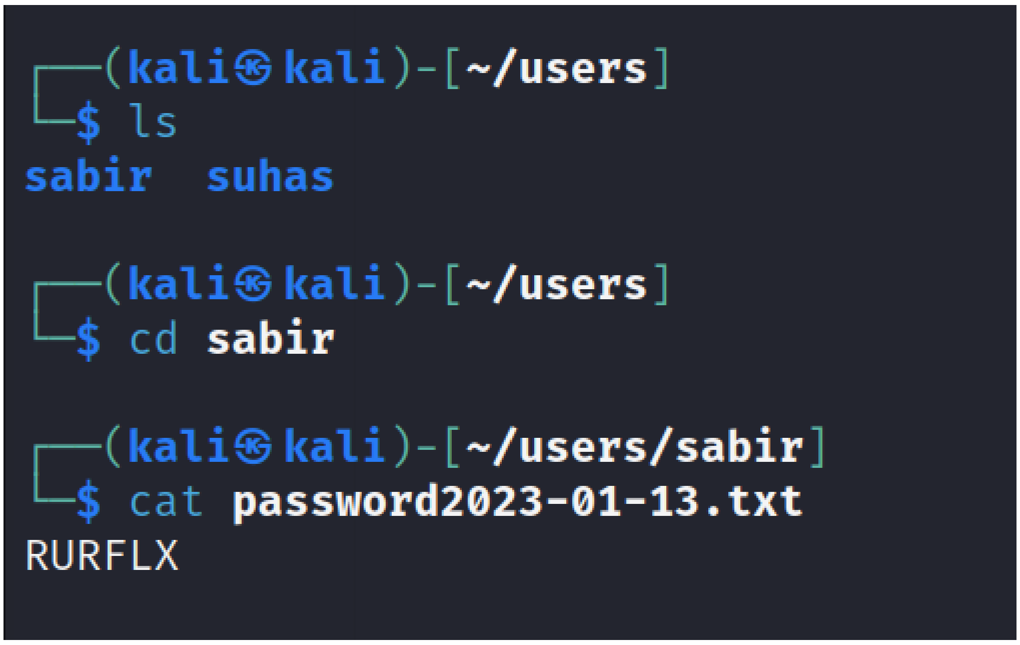
```
┌──(kali㉿kali)-[~]
└─$ ciphersec

What do you want to do next?
1. (CLI) Check ur Passw0rd! strength
2. (GUI) Check ur Passw0rd! strength
3. (CLI) Generate the unbreakable Passw0rd!
4. (GUI) Generate the unbreakable Passw0rd!
Enter your choice(1/2/3/4): 3
Enter Your Name: suhas
Enter Your Passphrase: cybersec
Generated Password:  GEUQVWKV
Password saved to users/suhas/password2023-01-13.txt.
```



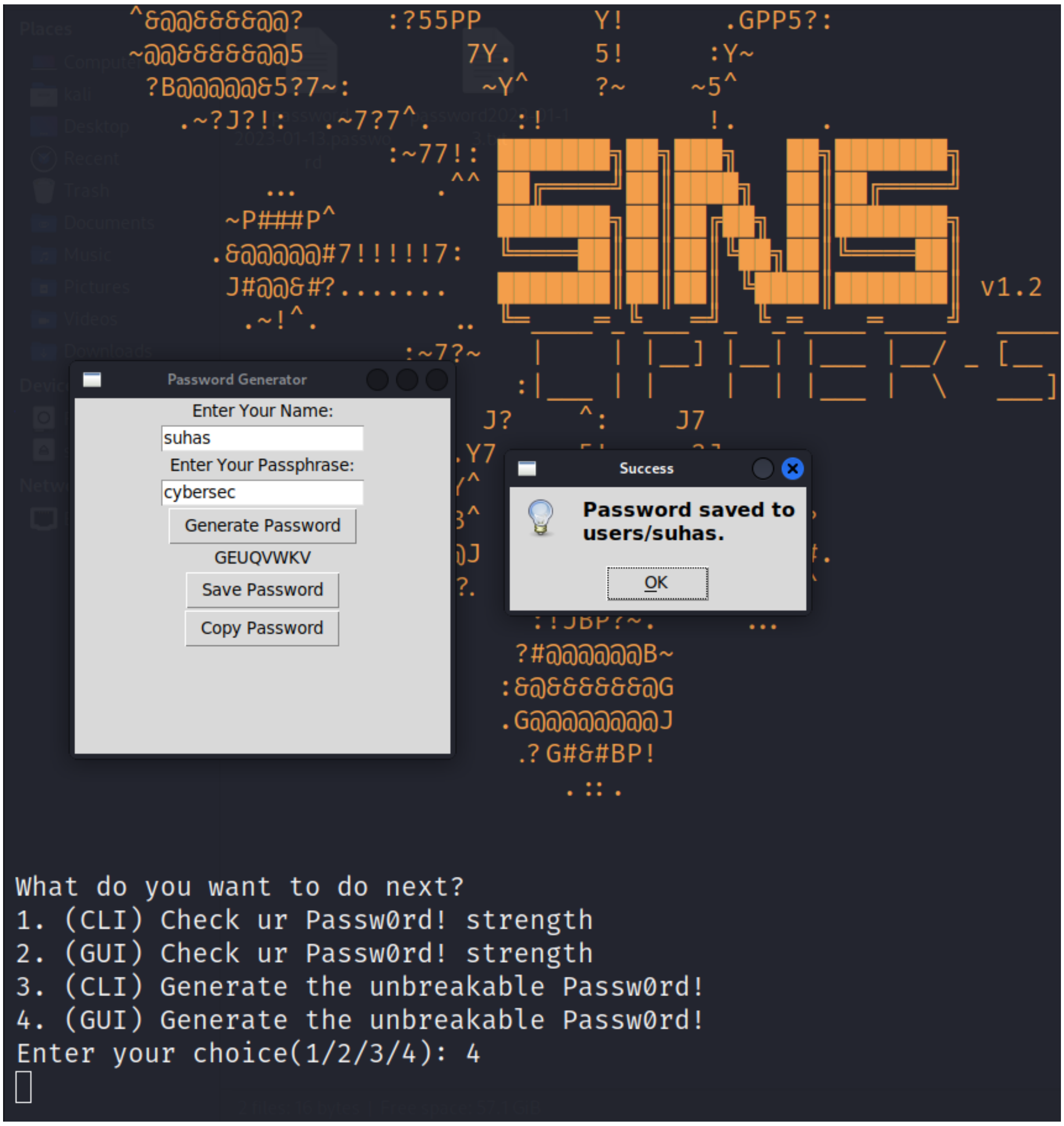
```
┌──(kali㉿kali)-[~/users/suhas]
└─$ cat password2023-01-13.txt
GEUQVWKV
```



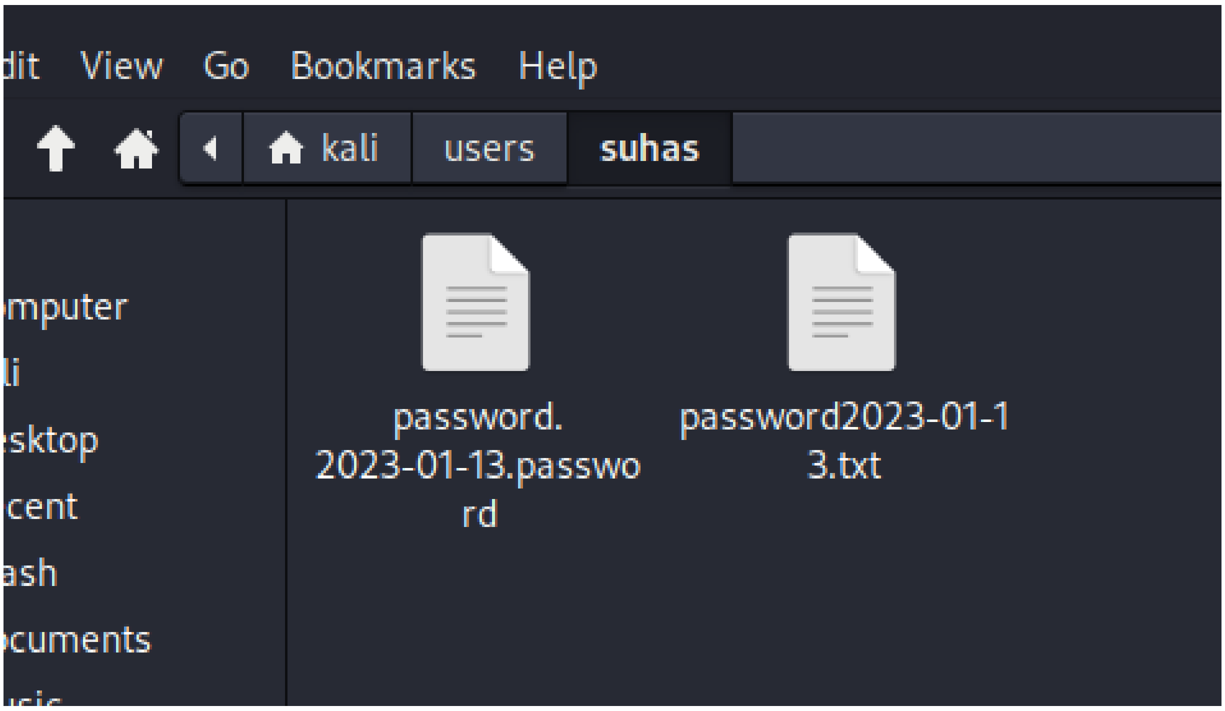
```
┌──(kali㉿kali)-[~]
└─$ ciphersec

       .---.
      /     \   _______ _       _                ______
      \.@-@./  (_______|_)     | |              / _____)
      /`\_/`\   _        _ ____ | |__  _____  ____( (____  _____  ____
     //  _  \\ | |      | |  _ \|  _ \| ___ |/ ___)\____ \| ___ |/ ___)
    | \     / || |______| | |_| | | | | ____| |    _____) ) ____( (___
   /`\_`> <`_/`\\_______)_|  __/|_| |_|_____)_|   (______/|_____)\____)
   \__/'---'\__/          |_|

What do you want to do next?
1. (CLI) Check ur Passw0rd! strength
2. (GUI) Check ur Passw0rd! strength
3. (CLI) Generate the unbreakable Passw0rd!
4. (GUI) Generate the unbreakable Passw0rd!
Enter your choice(1/2/3/4): 3
Enter Your Name: sabir
Enter Your Passphrase: nielit
Generated Password:  RURFLX
Password saved to users/sabir/password2023-01-13.txt.
```



```
┌──(kali㉿kali)-[~/users]
└─$ ls
sabir  suhas

┌──(kali㉿kali)-[~/users]
└─$ cd sabir

┌──(kali㉿kali)-[~/users/sabir]
└─$ cat password2023-01-13.txt
RURFLX
```



```
What do you want to do next?
1. (CLI) Check ur Passw0rd! strength
2. (GUI) Check ur Passw0rd! strength
3. (CLI) Generate the unbreakable Passw0rd!
4. (GUI) Generate the unbreakable Passw0rd!
Enter your choice(1/2/3/4): 4
```


dit
View
Go
Bookmarks
Help
kali
users
suhas
mputer
li
sktop
cent
ash
cuments
password.
2023-01-13.password
password2023-01-13.txt

CONCLUSION:

In conclusion, the SINS-CipherSec tool provides a comprehensive solution for securing online accounts by generating strong, unique passwords and storing them in a secure manner. The tool also allows users to assess the strength of their existing passwords and generate new, stronger passwords if necessary. The tool has a user-friendly CLI and GUI interface, making it easy for users to use. The results of the tool were evaluated and found to be highly effective in generating secure passwords and assessing password strength. Overall, the SINS-CipherSec tool is a valuable tool for anyone looking to secure their online accounts and protect their personal information.

FUTURE WORK

In the future, we plan to expand the password strength checker by incorporating more advanced algorithms that can detect patterns and common weaknesses in passwords. Additionally, we plan to integrate the tool with popular password managers to make it even easier for users to securely store and manage their passwords. We also plan to improve the user interface to make it more user-friendly and intuitive. Furthermore, we plan to add more security features such as two-factor authentication to provide an additional layer of security for our users.

will provide one stop solution for all password related solutions

will develop our project in more advanced manner and will implement it as web and mobile application

REFERENCES

https://devguide.python.org/
https://www.w3schools.com/python/
https://wmich.edu/arts-sciences/technology-password-tips#:~:text=Lowercase%20letters%3A%20a%2Dz,%22'.%3F%2F
https://docs.github.com/en
https://github.com/danielmiessler/SecLists/tree/master/Passwords
https://support.microsoft.com/en-us/windows/create-and-use-strong-passwords-c5cebb49-8c53-4f5e-2bc4-fe357ca048eb
https://support.google.com/accounts/answer/32040?hl=en
https://www.cisa.gov/uscert/ncas/tips/ST04-002
https://pages.nist.gov/800-63-3/sp800-63b.html
https://gteksd.wixsite.com/noob
https://github.com/GTekSD/SINS-CipherSec