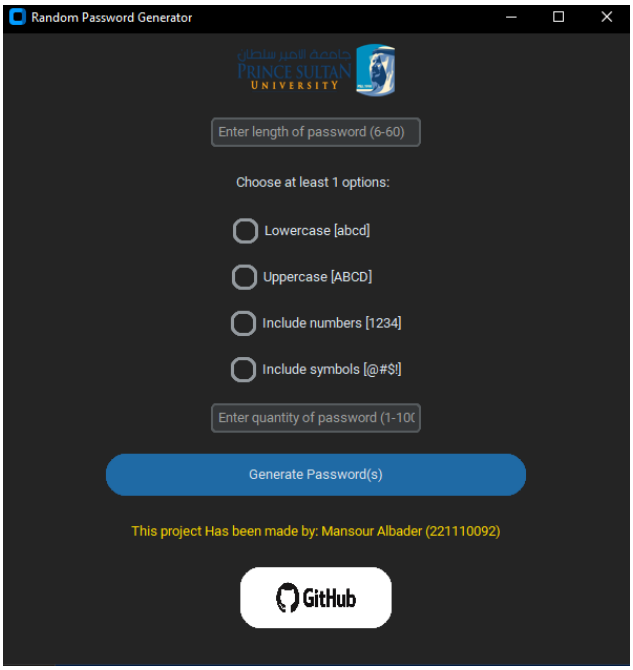
AUGUST 3, 2024

RANDOM PASSWORD GENERATOR



CYS-401 PROJECT

BY: Mansour Albader

MANSOUR ALBADER
221110092

Contents

Introduction

In today's interconnected digital landscape, cybersecurity plays a critical role in safeguarding sensitive information and protecting against unauthorized access. One fundamental aspect of cybersecurity is ensuring the strength and integrity of passwords. Random password generator, it helps create strong and unique passwords that are hard to crack using brute force attacks. By combining different types of characters like letters, numbers, and symbols, and varying the length of the password, random password generators greatly improve security of the password by using random characters.

Risks and threats

Security risks and threats. Weak passwords pose a security risk, as they are vulnerable to various types of attacks such as brute force attacks and dictionary attacks. These attacks aim to exploit vulnerabilities in password strength and gain access to user accounts, leading to unauthorized access.

Password strength

The strength of a password is determined by its complexity and randomness. Strong passwords are created using a mix of uppercase and lowercase letters, numbers, and symbols, making them harder to guess or crack through automated or manual methods.

Conclusion

In conclusion, random password generation is an essential technique of cybersecurity strategies, helping to mitigate common security risks, threats, attacks, and vulnerabilities associated with password-based authentication. By adopting robust password generation practices, individuals and organizations can enhance their resilience against cyber threats and safeguard sensitive information effectively.

Existing Methods

There are various online websites available that offer random password generation functionality. These websites may provide additional features such as password generator. All that to help users to generate a strong password that is hard to guess, by adding random characters, it will be complex for attacker to use brute force tools. In additional, many operating system offers a built-in software program that generate random passwords, also password manager software that help user to store their random password that is so hard to remember and allow users to generate a random password.

Previous Methods

Many organizations was enforce password policies requiring a mix of uppercase and lowercase letters only, any other characters will be optionally, to enhance password strength. This method makes the password easy to guess. Most of users always using their personal data in their password such as date of birth or their names. Before this was enough but, today, attackers build brute force tools that can easily guess password these passwords by using wordlists of passwords or try to find data that might users use in their passwords.

Substitute Methods

Today, passwords in not enough to secure users accounts, attackers find users passwords in multiple data bases that has been breaches and shared in dark web. Organization finds substitute methods to ensure secure users accounts, such as tow factor authentication that allow users to receive a code that has been generated for one time, to use it while signing in after using their password.

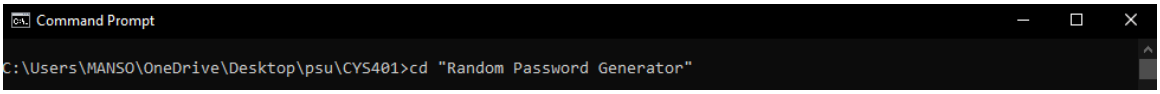
Security Incidents and Vulnerabilities

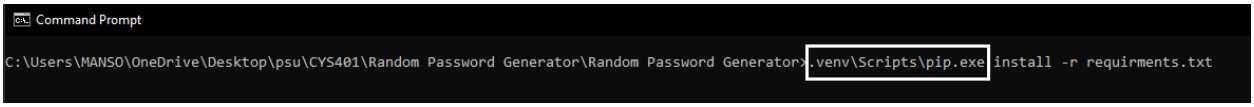
- Data breaches: Numerous high-profile data breaches have occurred due to weak or compromised passwords, leading to unauthorized access to sensitive information.
- Password reuse: Users often reuse passwords across multiple accounts, increasing the risk of widespread security breaches if one account is compromised.
- Phishing attacks: Attackers frequently use phishing emails to trick users into disclosing their passwords, exploiting human error rather than technical vulnerabilities.

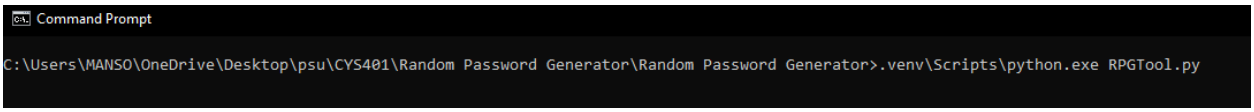
Implementation & requirements

Random password generator tool (RPGTool) is a software program has been programed using python language. RPGTool generate random passwords using random character chosen by the user such as (LowerCaseLatters, UpperCaseLatters, Numbers and Symbols).

Windows Installation steps:

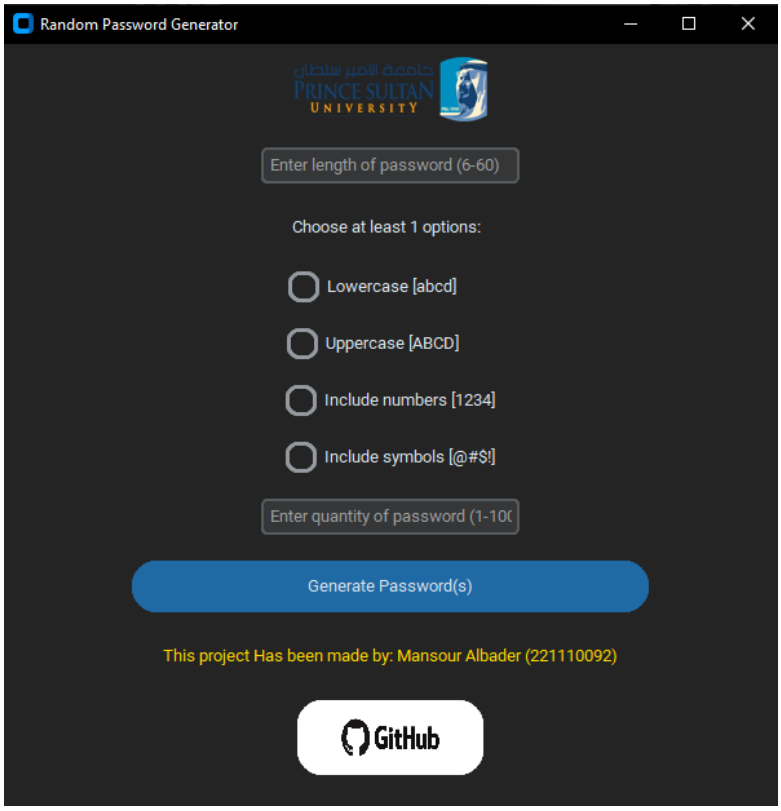
1. Visit https://github.com/MSecurity0/Random-Password-Generator
2. Download the files
3. Open cmd
4. cd dirName
5. 

6. Locate pip file in device c://example/pip.exe
7. pip.exe install -r requirments.txt
8. 

9. Run the tool using python
10. 

Linux Installation steps:

1. Open terminal
2. git clone https://github.com/MSecurity0/Random-Password-Generator
3. cd 'Random Password Generator'
4. pip install -r requirments.txt
5. python RGPTool.py



As it shows in the image, there is tow labels (length and quantity) and 4 choices (LowerCaseLatters, UpperCaseLatters, Numbers and Symbols). User can generate up to 100 passwords with length between 6-60 using the RGPTool.

(Source code explanation)

(Full source code algorthim.py)

(Full source code RPGTool.py)

Source Code

1) algorithm.py

a) Library's

```
import os.path
import customtkinter as ctk
import random
import string
import tkinter as tk
from tkinter import messagebox
from PIL import Image
from customtkinter import CTkLabel
import webbrowser
```

(Libraries needed to run the code has been imported in the top of the source code in algorithm.py)

b) PasswordGeneratorApp class

i) GUI

```
class PasswordGeneratorApp:
    def __init__(self, master):
        self.master = master
        master.geometry("600x600")
        master.title("Random Password Generator")
```

(This part of the code shows the startup resolution and the title of the tool in algorithm.py)

ii) PSU Logo

```
self.image_path=os.path.join(os.path.dirname(__file__),"images/logo.jpg")
self.image =
ctk.CTkImage(light_image=Image.open(self.image_path),size=(150,50))
self.image_label = CTkLabel(master=master, image=self.image,text='')
self.image_label.pack(side='top',pady=10)
```

iii) Password length entry and choose label

```
self.entry_length = ctk.CTkEntry(master,placeholder_text='Enter length of
password (6-60)',width=200)
self.entry_length.pack(side='top',pady=10)

self.label_options = ctk.CTkLabel(master, text="Choose at least 1 options:
")
self.label_options.pack(side='top',pady=10)
```

iv) Checkbox for (lowercase, uppercase, numbers and symbols)

```
self.lowercase_var = tk.BooleanVar()
self.lowercase_checkbox = ctk.CTkCheckBox(master,corner_radius=10, text="Lowercase [abcd]", variable=self.lowercase_var)
self.lowercase_checkbox.pack(side='top', pady=10)

self.uppercase_var = tk.BooleanVar()
self.uppercase_checkbox = ctk.CTkCheckBox(master,corner_radius=10, text="Uppercase [ABCD]", variable=self.uppercase_var)
self.uppercase_checkbox.pack(side='top',pady=10)

self.include_numbers_var = tk.BooleanVar()
self.include_numbers_checkbox = ctk.CTkCheckBox(master, corner_radius=10,text="Include numbers [1234]",
variable=self.include_numbers_var)
self.include_numbers_checkbox.pack(side='top',pady=10)

self.include_symbols_var = tk.BooleanVar()
self.include_symbols_checkbox = ctk.CTkCheckBox(master,corner_radius=10, text="Include symbols [@#$!]",
variable=self.include_symbols_var)
self.include_symbols_checkbox.pack(side='top',pady=10)

self.entry_quantity = ctk.CTkEntry(master,placeholder_text='Enter quantity of password (1-100)',width=200)
self.entry_quantity.pack(side='top',pady=10)
```

v) **Button to generate the password**

```
self.generate_button = ctk.CTkButton(master,corner_radius=32,
width=400,height=40,text="Generate Password(s)",
command=self.generate_passwords)
self.generate_button.pack(side='top',pady=10)
```

vi) **Tag and GitHub button**

```
self.label_image2 = ctk.CTkLabel(master, text_color='gold',
                                 text="This project Has been made by:
Mansour Albader (221110092)")
self.label_image2.pack(side='top', pady=10)


self.Gitimage_path = os.path.join(os.path.dirname(__file__),
"images/git.png")
self.Gitimage = ctk.CTkImage(light_image=Image.open(self.Gitimage_path),
size=(100, 50))
self.git_button =
ctk.CTkButton(master,fg_color='white',image=self.Gitimage,hover=None,
corner_radius=32, width=40,height=40,text="", command=self.openGit)
self.git_button.pack(side='top',pady=10)
```

c) Generate_password class

i) **Random small letters, capital letters and digits (using string lib)**

```
def generate_passwords(self):
    sLetters = string.ascii_lowercase
    cLetters = string.ascii_uppercase
    digit = string.digits
```

ii) **Defining all variables of random characters**

```
length = int(self.entry_length.get())
include_lowercase = self.lowercase_var.get()
include_uppercase = self.uppercase_var.get()
include_numbers = self.include_numbers_var.get()
include_symbols = self.include_symbols_var.get()
quantity = int(self.entry_quantity.get())
```

iii) **Generating algorithm**

```
characters = ''
if include_lowercase:
    characters += sLetters
if include_uppercase:
    characters += cLetters
if include_numbers:
    characters += digit
if include_symbols:
    characters += string.punctuation

passwords = []
if 1 <= quantity <= 100:
    i = 0
    for _ in range(quantity):
        i += 1
        if 6 <= length <= 60:
            password = ''.join(random.choice(characters) for _ in range(length))
            passwords.append(str(i)+': '+password+'\n----------------')
        else:
            s = "password length must be between 6 and 60"
            passwords.append(s)
else:
    s = "quantity length must be between 1 and 100"
    passwords.append(s)
```

iv) **Pop-up windows for the passwords generated or errors**

```
password_str = "\n".join(passwords)
messagebox.showinfo("Generated Passwords", password_str)
```

v) **Open GitHub definition [Tag and GitHub]**

```
def openGit(self):
    return webbrowser.open('https://github.com/MSecurity0/Random-Password-
Generator')
```

2) RPGTool.py

a) Libraries

```
import algorithm
import customtkinter as ctk
```

b) Run the tool

```
root = ctk.CTk()
app = algorithm.PasswordGeneratorApp(root)
root.mainloop()
```

Full source code algorithm.py

```
import os.path
import customtkinter as ctk
import random
import string
import tkinter as tk
from tkinter import messagebox
from PIL import Image
from customtkinter import CTkLabel
import webbrowser


class PasswordGeneratorApp:
    def __init__(self, master):
        self.master = master
        master.geometry("600x600")
        master.title("Random Password Generator")

        self.image_path = os.path.join(os.path.dirname(__file__),
"images/logo.jpg")
        self.image = ctk.CTkImage(light_image=
Image.open(self.image_path),size=(150,50))
        self.image_label = CTkLabel(master=master, image=self.image,text='')
        self.image_label.pack(side='top',pady=10)

        self.entry_length = ctk.CTkEntry(master,placeholder_text='Enter
length of password (6-60)',width=200)
        self.entry_length.pack(side='top',pady=10)

        self.label_options = ctk.CTkLabel(master, text="Choose at least 1
options:  ")
        self.label_options.pack(side='top',pady=10)

        self.lowercase_var = tk.BooleanVar()
        self.lowercase_checkbox = ctk.CTkCheckBox(master,corner_radius=10,
text="Lowercase [abcd]          ", variable=self.lowercase_var)
        self.lowercase_checkbox.pack(side='top', pady=10)

        self.uppercase_var = tk.BooleanVar()
        self.uppercase_checkbox = ctk.CTkCheckBox(master,corner_radius=10,
text="Uppercase [ABCD]          ", variable=self.uppercase_var)
        self.uppercase_checkbox.pack(side='top',pady=10)

        self.include_numbers_var = tk.BooleanVar()
        self.include_numbers_checkbox = ctk.CTkCheckBox(master,
corner_radius=10,text="Include numbers [1234]",
variable=self.include_numbers_var)
        self.include_numbers_checkbox.pack(side='top',pady=10)

        self.include_symbols_var = tk.BooleanVar()
        self.include_symbols_checkbox =
ctk.CTkCheckBox(master,corner_radius=10, text="Include symbols [@#$!]",
variable=self.include_symbols_var)
        self.include_symbols_checkbox.pack(side='top',pady=10)
```

```
        self.entry_quantity = ctk.CTkEntry(master,placeholder_text='Enter
quantity of password (1-100)',width=200)
        self.entry_quantity.pack(side='top',pady=10)

        self.generate_button = ctk.CTkButton(master,corner_radius=32,
width=400,height=40,text="Generate Password(s)",
command=self.generate_passwords)
        self.generate_button.pack(side='top',pady=10)

        self.label_image2 = ctk.CTkLabel(master, text_color='gold',
                                         text="This project Has been made by:
Mansour Albader (221110092)")
        self.label_image2.pack(side='top', pady=10)


        self.Gitimage_path = os.path.join(os.path.dirname(__file__),
"images/git.png")
        self.Gitimage =
ctk.CTkImage(light_image=Image.open(self.Gitimage_path), size=(100, 50))
        self.git_button =
ctk.CTkButton(master,fg_color='white',image=self.Gitimage,hover=None,
corner_radius=32, width=40,height=40,text="", command=self.openGit)
        self.git_button.pack(side='top',pady=10)



    def generate_passwords(self):
        sLetters = string.ascii_lowercase
        cLetters = string.ascii_uppercase
        digit = string.digits

        length = int(self.entry_length.get())
        include_lowercase = self.lowercase_var.get()
        include_uppercase = self.uppercase_var.get()
        include_numbers = self.include_numbers_var.get()
        include_symbols = self.include_symbols_var.get()
        quantity = int(self.entry_quantity.get())


        characters = ''
        if include_lowercase:
            characters += sLetters
        if include_uppercase:
            characters += cLetters
        if include_numbers:
            characters += digit
        if include_symbols:
            characters += string.punctuation
```

```
        passwords = []
        if 1 <= quantity <= 100:
            i = 0
            for _ in range(quantity):

             i += 1
             if 6 <= length <= 60:
                password = ''.join(random.choice(characters) for _ in range(length))
                passwords.append(str(i)+': '+password+'\n------------------')

                else:
                    s = "password length must be between 6 and 60"
                    passwords.append(s)
                    break
        else:
            s = "quantity length must be between 1 and 100"
            passwords.append(s)

        password_str = "\n".join(passwords)
        messagebox.showinfo("Generated Passwords", password_str)

    def openGit(self):
        return webbrowser.open('https://github.com/MSecurity0/Random-Password-Generator')
```

Full source code RPGTool.py

```
import algorithm
import customtkinter as ctk

root = ctk.CTk()
app = algorithm.PasswordGeneratorApp(root)
root.mainloop()
```