SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

SCHOOL OF COMPUTING

DEPARTMENT OF DATASCIENCE AND BUSINESS SYSTEMS

18CSC304J COMPILER DESIGN

MINI PROJECT REPORT

Title – Random Password Generator

NAME: YUVRAJ SINGH CHAUHAN, ARNAV KUMAR

REGISTER NUMBER: RA1911027010058, RA1911027010040

DEPARTMENT: B.TECH

SPECIALIZATION: CSE BIG DATA ANALYTICS

SEMESTER: VI

CONTENT PAGE

INTRODUCTION

Text based username-password is the most commonly employed authentication mechanism in many multiuser environments. These multiuser applications, while registering users to their application, some applications allow users to create password their own and others generate random password and supply to users. Various surveys have shown users created passwords are less secure than system generated passwords. Most user created passwords can be found in common password lists on internet. The user created passwords can be guessed easily, with a bit of social engineering like user's personal information or type of application. System generated passwords cannot be guessed easily and have no relevance with the user’s personal information and type of application but are hard to remember.

Text based password authentication systems involve a tradeoff between security and memorability of passwords. Some passwords are easy to remember but also easy to guess for an adversary. Random passwords are hard to remember and hard to crack because they are made up of arbitrary sequence of characters. Several studies have examined how password composition policies affect users. In some studies, it is revealed that password composition policies influence the predictability of passwords and how well they affect the user behaviour and sentiments. Their results demonstrate that successfully creating a password is significantly more difficult under stricter password composition policies. They measured how many people failed at least once to create an acceptable password and further observed how users deal with it.

SYNOPSIS

Passwords are a real security threat. Over 80% of hacking-related breaches are due to weak or stolen passwords, a recent report shows . So if you want to safeguard your personal info and assets, creating secure passwords is a big first step. And that’s where Random Password Generator can help. Impossible-to-crack passwords are complex with multiple types of characters (numbers, letters, and symbols). Making your passwords different for each website or app also helps defend against hacking.

A random password generator is software program or hardware device that takes input from a random or pseudo-random number generator and automatically generates a password. Random passwords can be generated manually, using simple sources of randomness such as dice or coins, or they can be generated using a computer.

This random password generator is built around lexical analysis using python where we are converting characters into tokens and concatenating them to make a random password for the user

LANGUAGE AND MODULES

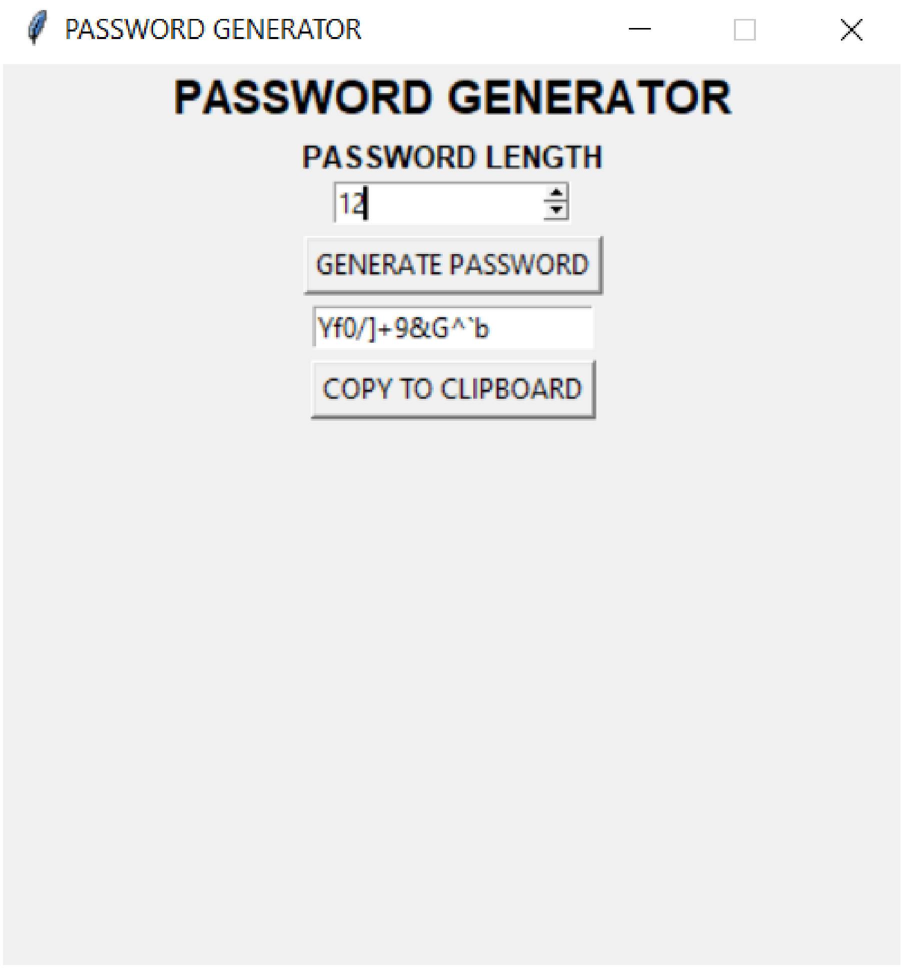
- Python
- String Module
- Random Module
- Tkinter Module

SOURCE CODE

```
from tkinter import *
import random, string
import pyperclip
```

```
root =Tk()
root.geometry("400x400")
root.resizable(0,0)
root.title("PASSWORD GENERATOR")

heading = Label(root, text = 'PASSWORD GENERATOR' , font ='arial 15
bold').pack()

pass_label = Label(root, text = 'PASSWORD LENGTH', font = 'arial 10
bold').pack()
pass_len = IntVar()
length = Spinbox(root, from_ = 8, to_ = 32 , textvariable = pass_len ,
width = 15).pack()

pass_str = StringVar()

def Generator():
    password = ''
    for x in range (0,4):
        password =
random.choice(string.ascii_uppercase)+random.choice(string.ascii_lower
case)+random.choice(string.digits)+random.choice(string.punctuation)
    for y in range(pass_len.get()- 4):
        password = password+random.choice(string.ascii_uppercase +
string.ascii_lowercase + string.digits + string.punctuation)
    pass_str.set(password)

Button(root, text = "GENERATE PASSWORD" , command = Generator
).pack(pady= 5)
Entry(root , textvariable = pass_str).pack()

def Copy_password():
    pyperclip.copy(pass_str.get())

Button(root, text = 'COPY TO CLIPBOARD', command =
Copy_password).pack(pady=5)

root.mainloop()
```

OUTPUT


PASSWORD GENERATOR
PASSWORD GENERATOR
PASSWORD LENGTH
12
GENERATE PASSWORD
Yf0/]+9&G^`b
COPY TO CLIPBOARD

CONCLUSION

Complex password composition policies and policies that require password must be changed after a period of time happens to be major obstacle for users. The proposed technique can assist system administrators in creating secure and memorable passwords for users with desired complex password composition policies. The generated password along with helping information (random word, random position string and random character string) will be sent to users.

This technique gives several benefits to users such as security, and confidentiality. The password generated using proposed technique is more secure because it is chosen from a large distribution of passwords and is stronger than user created passwords. The proposed technique causes more Confidentiality because in this technique, distinct passwords are given to users on different applications.

If an application is compromised then rest of all are protected. Future work includes determining the memorability of the generated password. Intuitively, it can be said that the passwords generated using proposed technique are more memorable than pure random passwords.