A picture containing graphic design, graphics, poster, art

Description automatically generated

Instituto Politécnico Nacional

Unidad Profesional Interdisciplinaria en Ingeniería y Tecnologías Avanzadas

Propedéutico Programación

Reporte

Proyecto Final: Generador de Contraseñas

Alumno: Carmona Bellido Marco Tulio

Profesor: Dr. Miguel Félix Mata Rivera

Resumen.

Este proyecto presenta una propuesta de solución a un programa generador de contraseñas. Haciendo uso de sockets en Python se crea un servidor al cual los clientes le solicitan generar una contraseña a través de ciertos parámetros y el servidor responde enviando la contraseña generada.

Marco Teórico.

Generador de contraseñas

Es un programa que es capaz de generar contraseñas de forma inmediata y aleatoria o pseudoaleatoria a partir de un pequeño numero de valores de entrada [1]. Un generador de contraseñas común tiene los siguientes componentes:

* Un *grupo de valores de entrada* los cuales se usan para determinar las características de esta dependiendo del uso que tendrá. Estos valores pueden ser: el tamaño de la contraseña, el almacenamiento (online o local), características especificadas por el sitio o cuenta para el cual se está creando la contraseña.
* Una *función de generación de contraseñas* que combine los valores de entrada para generar una contraseña adecuada. La función debe adaptarse a los requerimientos establecidos por el sitio web, aplicación o programa que solicita la contraseña.
* Un *método de exportar la contraseña* generada de forma que esta pueda ser usada para el usuario, aplicación, etc. (por ejemplo, mostrara la contraseña al usuario.)

Función aleatoria en Python

En Python existes dos funciones principales para generar números aleatorios, la función random() y la función secrets().

La función random() genera números pseudoaleatorios y no se recomienda para su uso con fines de seguridad, para esto ultimo se sugiere usar la función secrets() la cual es capas de generar numero aleatorios seguros para trabajar con secretos criptográficos. La función random() genera numeros aleatorios a partir de fuentes proporcionadas por el sistema operativo, la información proporcionada por el sistema operativo es lo suficientemente impredecible para poder ser utilizada en aplicaciones criptográficas, sin embargo la calidad exacta de esta depende de[ende de su implementación en el sistema operativo mismo, es decir no es lo mismo la función de Windows que devuelve variables aleatorias del sistema (BCryptGenRandom()) que la función de Linux (getrandom()). [2].

Por su parte la función secrets() provee acceso a la fuente más segura de aleatoriedad que proporciona el sistema operativo sobre el cual se esta ejecutando el programa. Esta función también es capaz de generar tokens seguros, los cuales pueden ser usados en aplicaciones de restablecimiento de contraseñas por ejemplo generar URLs que sean difíciles de adivinar. [3]

Descripción del problema

El problema consiste en crear un socket servidor en un lenguaje arbitrario que sirva para generar contraseñas basadas en un pequeño numero de valores de entrada. A su vez se debe crear un cliente que se conecte a dicho servidor y sea capaz de enviar los parámetros de entrada para que el servidor genere la contraseña, el cliente debe recibir la contraseña y mostrársela al usuario, por último, se requiere de un segundo cliente el cual debe ser capaz de solicitar la contraseña generada al servidor y debe permitir guardarla junto con las demás contraseñas que se hayan solicitado para algún posterior uso.

Propuesta de solución

El lenguaje de programación elegido para abordar el problema fue Python ya que ofrece muchas facilidades en cuestión de paquetes que permiten hacer una implementación más rápida de varias funciones, tales como generar números random, manejo de archivos y creación de GUI.

1. Servidor

El script de Python del servidor (pswdServer.py) emplea los paquetes de socket, threading y random. En el marco teórico se menciono que para generar números aleatorios seguros se debe usar el paquete *secrets* en lugar de *random,* sin embargo el uso de *random* en este programa se debe a un problema de instalación del paquete *secrets* en la maquina con la que se programó, además de que este requiere de paquete depende de tener otros paquetes previamente instalados, por lo que para este caso especifico se emplea *random*, sin embargo si se quiere implementar para aplicaciones de seguridad el paquete debe ser reemplazado necesariamente. Los paquetes de socket y threading permiten al programa comunicarse mediante sockets con otros programas y hacerlos con más de uno simultáneamente.

El siguiente programa describe de manera general el funcionamiento del programa:

A picture containing text, diagram, plan, technical drawing

Description automatically generated

Figura 1. descripción del funcionamiento del servidor.

El programa espera las conexiones de los clientes, cada que una conexión es establecida con éxito se crea una instancia de tipo NweClientSocketHandler que se encargara de la comunicación con el cliente, esta es la que se encarga de escuchar al cliente y responder en base a los datos recibidos, si recibe los parámetros de configuración correctos entonces se llamara a la función pass\_generate(), la cual genera la contraseña usando como elementos los principales caracteres que son solicitados por las aplicaciones o sitios web para uso en contraseñas, se usa un alfabeto alfanumérico y algunos caracteres especiales (los más comúnmente permitidos). La contraseña generada se envía al cliente que la solicito como respuesta.

El servidor no tiene la capacidad de almacenar las contraseñas generadas, es decir solo guarda la última generada, sin embargo, esto no representa un problema ya que los scripts de los clientes si permiten guardar las contraseñas en archivos locales en la computadora.

Una porción del código se muestra a continuación, sin embargo, el código completo se encuentra en el siguiente repo de GitHub <https://github.com/TulioCarmona/PasswordGenerator.git>

A screenshot of a computer program

Description automatically generated with low confidence

Por ultimo se muestran un par de capturas del programa siendo ejecutado. Este programa solo debe ser ejecutado, es decir, no le solicita alguna entrada al usuario y además funciona automáticamente, únicamente se puede apreciar en la terminar, cunado está esperando conexión con un cliente y muestra los datos de la conexión realizada con los diferentes clientes.

A picture containing text, electronics, screenshot, software

Description automatically generated

Figura 2. Terminal de Windows antes de ejecutar el programa del servidor.

A computer screen with white text

Description automatically generated with low confidence

Figura 3. Inicio del servidor, esperando por conexiones.

A screenshot of a computer program

Description automatically generated with medium confidence

Figura 4. Servidor conectado a dos clientes.

1. Cliente 1

El primer cliente se encarga de permitir al usuario configurar los parámetros de la contraseña (longitud, numero de caracteres especiales, mayúsculas, minúsculas y números). Con el fin de facilitar la tarea al usuario se diseño una GUI basada en una versión simplificada de gestores de contraseñas tales como LastPass [4] y NordPass[5].

La Interfaz gráfica se diseño usando el paquete PyQt5 [6] de Python (figura 5), además se emplearon los paquetes de sys y socket para la comunicación con el servidor y el manejo de archivos.

A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated

Figura 5. GUI del cliente para generar las contraseñas.

El cliente se conecta con el servidor, al hacerlos se muestra la GUI, la cual contiene elementos como sliders, botones y checkboxes y spinboxes que hacen que establecer los parámetros sea muy fácil. Como se muestra en la figura 5, para habilitar alguno de los caracteres especiales es necesario marcar la casilla con su nombre, así se habilitará un spinbox para establecer el numero de este tipo de caracteres que aparecerá en la contraseña final, si el checkbox es deshabilitado nuevamente, el numero previamente establecido será ignorado y devuelto a un valor de cero.

Si por error la suma de todos los caracteres especiales sobrepasa la longitud de la contraseña establecida el programa marcara una alerta al solicitar dicha contraseña y solicitara corrija los datos (ver figura 6).

A screenshot of a computer

Description automatically generated

Figura 6. Error al generar la contraseña por parámetros erróneos.

Si los valores de configuración son correctos y se presiona el botón de generar, entonces la contraseña se mostrar justo debajo del botón de generara junto con un botón de guardar (figura 7). A si mismo la contraseña generada se podrá modificar a criterio del usuario, agregar, quitar o cambiar caracteres, sin embargo, para que la contraseña modificada se actualice en el servidor se debe presionar el botón de guardar.

A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated

Figura 7. Generación exitosa de la contraseña

Por último, el botón de guardar abrirá una ventana del explorador de archivos permitiendo al usuario elegir la ubicación y nombre del archivo en el que se guardará la contraseña.

A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated

Figura 8. Selección de la ubicación y nombre para guardar la contraseña.

El cliente puede solicitar tantas configuraciones de contraseñas tantas como desee y modificarlas a criterio propio, de igual forma puede guardar todas las contraseñas que desee.

1. Cliente 2

El segundo cliente posee una interfaz muy sencilla, esto debido a que su única función es solicitad la contraseña generada por el otro cliente al servidor.

Para este programa se usaron las mismas dependencias que con el cliente 1. La interfaz gráfica inicialmente muestra únicamente un botón que dice “solicitar contraseña”.

A screenshot of a computer

Description automatically generated with medium confidence

Figura 9. Ejecución del script del cliente 2.

Si dicho botón es presionado el cliente se comunicará con el servidor indicándole que desea saber cual es la ultima contraseña generada, el servidor responderá y la contraseña se mostrará justo debajo del botón junto con un nuevo botón de guardar. En este caso la contraseña no se puede editar solo visualizar y guardar, además al presionar el botón de guardar el programa mostrará una ventana emergente indicando la dirección de donde se guardo el archivo con el historial de contraseñas ya que cada que guarda una contraseña lo hace un mismo archivo donde se encuentran guardadas contraseñas previamente solicitadas. La dirección de dio historial será la misma que desde donde se ejecuta el programa, aun así, el programa mostrará la ruta completa de guardado para que el usuario no tenga problemas.

A screenshot of a computer

Description automatically generated with medium confidence

Figura 10. Solicitud de la contraseña al servidor.

A screenshot of a computer screen

Description automatically generated with medium confidence

Figura 11. Mensaje de guardado exitoso de la contraseña en la misma ruta que el programa.

Si uno busca el archivo con el historial de las contraseñas, el cual por defecto se llama pswdHistory.txt, al abrirlo se podrá observar una lista de la siguiente manera:

A screenshot of a computer

Description automatically generated

Figura 12. Historial de contraseñas.

Como se ve en la figura 12 cada contraseña se encuentra acompañada de la fecha y hora en la que fue guardada.

Trabajo a futuro

1. Todos los programas fueron probados localmente, en un futo se ha de verificar que estos funcionen correctamente en una red distinta.
2. Tal como se planteo al inicio de la propuesta de solución se debe remplazar el uso de random() con secrets() para que este programa pueda usarse con fines de seguridad reales.
3. Los caracteres con los que se pueden crear la contraseña se encuentran previamente definidos y son los más comunes utilizados para dicho fin, considerando la región occidental en la que nos encontramos, por ejemplo, en varios sitios no se aceptan contraseñas con caracteres especiales, únicamente con letras y números. Se podría probar agregando caracteres de otros alfabetos, sin embargo, para poder trabajar correctamente con ellos se debe establecer el tipo de codificación correcta, de otra forma el programa marcaria error al enviar o guardar dichos caracteres.

Referencias.

[1] Al Maqbali, F., Mitchell, C.J. (2016). Password Generators: Old Ideas and New. In: Foresti, S., Lopez, J. (eds) Information Security Theory and Practice. WISTP 2016. Lecture Notes in Computer Science(), vol 9895. Springer, Cham. <https://doi-org.bibliotecaipn.idm.oclc.org/10.1007/978-3-319-45931-8_16>

[2] Python Software Foundation. Interfases misceláneas del sistema operativo. [Online]. Disponible en: <https://docs.python.org/es/3/library/os.html#os.urandom>

[3] Python Software Foundation. Secrets - Genera números aleatorios seguros para trabajar con secretos criptográficos. [Online]. Disponible es: <https://docs.python.org/es/3/library/secrets.html#module-secrets>

[4] LastPass Todos los derechos reservados. HERRAMIENTA DE GENERACIÓN DE CONTRASEÑAS. [Online]. Disponible en: <https://www.lastpass.com/es/features/password-generator>

[5] Nord Security Todos los derechos reservados (2023, febrero 22). NordPass password manager: Next-level password security. [Online]. Disponible en: <https://nordvpn.com/es/blog/nordpass-password-manager/>

[6] Riverbank Computing. What is PyQt?. [Online]. Disponible en: <https://www.riverbankcomputing.com/software/pyqt/>