**Asigna y reasigna variables en Python**

Anteriormente, exploraste las variables y cómo asignarlas y reasignarlas en Python. En esta lectura, ampliarás tu comprensión sobre estos temas. También, aprenderás sobre la práctica general de nombrar variables para evitar errores de sintaxis y mejorar la legibilidad del código.

**¿Qué son las variables?**

En un lenguaje de programación, una **variable** es un contenedor que almacena datos. Es una ubicación de almacenamiento con nombre en la memoria de una computadora que puede contener un valor. Almacena los datos en un tipo de dato específico, como entero, cadena o booleano. El valor que se almacena en una variable puede cambiar.

Puedes pensar en las variables como cajas con etiquetas. Aunque cambies el contenido de una caja, su etiqueta permanece igual. De manera similar, cuando cambias el valor almacenado en una variable, el nombre de esta sigue siendo el mismo.

Los analistas de seguridad que trabajan en Python utilizarán distintas variables. Algunos ejemplos incluyen variables para intentos de inicio de sección, listas de permisos y direcciones.

**Trabajar con variables**

En Python, es importante saber cómo asignar variables y cómo reasignarlas.

**Asignación y reasignación de variables**

Si deseas crear una variable llamada **username** y asignarle un valor de **"nzhao"**, coloca la variable a la izquierda del signo igual y su valor a la derecha:

**# Assign 'username'**

**username = "nzhao"**

Si luego restableces este nombre de usuario a **"zhao2"**, seguirás refiriéndote a ese contenedor de variable como **username**.

**# Reassign 'username'**

**username = "zhao2"**

Aunque el contenido haya cambiado de **"nzhao"** a **"zhao2"**, la variable **username** seguirá siendo la misma.

**Nota:** Debes colocar **"nzhao"** y **"zhao2"** entre comillas porque son cadenas. Python asigna automáticamente a una variable su tipo de datos cuando se ejecuta. Por ejemplo, cuando la variable **username** contiene la cadena **"nzhao"**, se le asigna un tipo de datos de cadena.

**Asignar variables a otras variables**

Mediante un proceso similar, también puedes asignar variables a otras variables. En el siguiente ejemplo, la variable **username** se asigna a una nueva variable **old\_username**:

**# Assign a variable to another variable**

**username = "nzhao"**

**old\_username = username**

Como **username** contiene el valor de cadena de **"nzhao"** y **old\_username** contiene el valor de **username**, **old\_username** ahora contiene el valor de **"nzhao"**.

**Combinar todo**

El siguiente código demuestra cómo se puede actualizar un nombre de usuario. A la variable **username** se le asigna un valor inicial, el cual luego se almacena en una segunda variable llamada **old\_username**. Después de esto, a la variable **username** se le reasigna un nuevo valor. Puedes ejecutar este código para obtener un mensaje sobre el nombre de usuario anterior y el nombre de usuario actual:

1

2

3

4

5

username = "nzhao"

old\_username = username

username = "zhao2"

print("Previous username:", old\_username)

print("Current username:", username)

EjecutarRestablecer

**Prácticas recomendadas para nombrar variables**

Puedes nombrar una variable casi como quieras, pero hay algunas pautas que debes seguir para asegurar una sintaxis correcta y evitar errores:

* Usa solo letras, números y guiones bajos en los nombres de las variables. Ejemplos válidos: **date\_3**, **username**, **interval2**
* Comienza el nombre de una variable con una letra o un guion bajo. No empieces con un número. Ejemplos válidos: **time**,  **\_login**
* Recuerda que los nombres de variables en Python distinguen entre mayúsculas y minúsculas. Todas estas son variables diferentes: **time**, **Time**, **TIME**, **timE**.
* No utilices las palabras clave o funciones integradas de Python para nombres de variables. Por ejemplo, las variables no deben llamarse **True**, **False**, o **if**.

Además, debes seguir estas pautas de estilo para que tu código sea más fácil de leer y comprender:

* Separa dos o más palabras con guiones bajos. Ejemplos válidos: **login\_attempts***,* **invalid\_user***,* **status\_update**
* Evita variables con nombres similares. Estas variables podrían confundirse fácilmente entre sí: **start\_time**, **starting\_time**, **time\_starting**.
* Evita nombres innecesariamente largos para las variables. Por ejemplo, no des a las variables nombres como **variable\_that\_equals\_3**.
* Los nombres deben describir los datos y no ser palabras aleatorias. Ejemplos válidos: **num\_login\_attempts**, **device\_id**, **invalid\_usernames**

**Nota**: Se recomienda usar guiones bajos para separar varias palabras en las variables, pero otra convención que puedes encontrar es escribir con mayúscula la primera letra de cada palabra, excepto la primera palabra. Ejemplo: **loginAttempt**

**Conclusiones clave**

Es importante que los analistas de seguridad conozcan muy bien las variables. Estas actúan como contenedores de datos. Se les asignan valores y también se les puede reasignar otros valores o variables. Recordar las mejores prácticas para nombrar variables resulta muy útil para crear un código más funcional y fácilmente legible.