LAB-012-Exemplar_Import and parse a text file

September 20, 2024

1 Modelo: Importa y analiza un archivo de texto

Introducción

A menudo, los registros de seguridad se almacenan en archivos de texto. Para analizarlos en estos archivos, los analistas de seguridad tienen que importarlos y analizarlos. Python tiene algunas funciones que son útiles para estas tareas, lo que permite a los analistas acceder de manera eficiente a la información de los archivos de texto.

En este laboratorio, practicarás el uso de funciones y otras sintaxis en Python para importar y analizar archivos de texto.

Consejos para completar este laboratorio

Mientras recorres este laboratorio, ten en cuenta los siguientes consejos:

— `### TU CÓDIGO AQUÍ ###` indica dónde debes escribir el código. Asegúrate de reemplazarlo con tu propio código antes de ejecutar la celda de código. — Siéntete libre de abrir las pistas para obtener información adicional a medida que trabajas en cada tarea. — Para ingresar tu respuesta a una pregunta, haz doble clic en la celda de markdown para editar. Asegúrate de reemplazar "[Haz doble clic para ingresar aquí tus respuestas.]" con tu propia respuesta. — Puedes guardar tu trabajo manualmente con un clic en Archivo y luego en Guardar en la barra de menú en la parte superior del cuaderno. — Puedes descargar tu trabajo localmente con un clic en Archivo y luego en Descargar, luego puedes especificar el formato de archivo que prefieras en la barra de menú en la parte superior del cuaderno.

1.1 Situación hipotética

En este laboratorio, trabajarás como analista de seguridad. Eres responsable de preparar un archivo de registro de seguridad para el análisis y crear un archivo de texto con direcciones IP a las que se les permite acceder a información restringida.

Tarea 1 En esta tarea, importarás un archivo de texto de registro de seguridad y lo almacenarás como una cadena para prepararlo para el análisis.

En Python, una sentencia `with` se suele utilizar en el manejo de archivos para abrirlos y luego cerrarlos automáticamente después de leerlos.

Se te proporciona una variable llamada `import_file` que contiene el nombre del archivo de registro que deseas importar. Comienza escribiendo la primera línea de la sentencia `with` en la siguiente celda de código. Utiliza la función `open()`, estableciendo el segundo parámetro en `"r"`. Ten en

cuenta que la ejecución de este código producirá un error, ya que solo contendrá la primera línea de la sentencia `with`; completarás esta sentencia `with` en la tarea después de esto. Asegúrate de reemplazar `### TU CÓDIGO AQUÍ ###` con tu propio código.

[1]:
```
# Asigna a `import_file` el nombre del archivo de texto que contiene el archivo␣
↪de registro de seguridad

import_file = "data/login.txt"

# La primera línea de la sentencia `with`
# Utiliza `open()` para importar el archivo de registro de seguridad y␣
↪almacenarlo como una cadena

with open(import_file, "r") as file:
```

```
       [0;36m  File [0;32m"<ipython-input-1-92ba46900c1d>"[0;36m, line [0;
↪32m8[0m
    [0;31m    with open(import_file, "r") as file:[0m
    [0m                                            ^[0m
    [0;31mSyntaxError[0m[0;31m:[0m unexpected EOF while parsing
```

Pista 1

La función `open()` en Python te permite abrir un archivo.

Como primer parámetro, toma el nombre del archivo (o una variable que contiene el nombre del archivo). Como segundo parámetro, toma una cadena que indica cómo se debe gestionar el archivo.

Pasa la letra `"r"` como segundo argumento cuando quieras leer el archivo.

Pista 2

La variable `import_file` contiene el nombre del archivo que deseas abrir, por lo que debería ser el primer argumento que pases a la función `open()`.

Dado que también quieres leer el contenido del archivo, debes pasar `"r"` como segundo argumento.

Asegúrate de usar una coma (`,`) para separar a ambos.

1.2 Tarea 2

Ahora, utilizarás el método `.read()` para leer el archivo importado y almacenarás el resultado en una variable llamada `text`. Después, muestra el `texto` y explora lo que contiene al ejecutarse la celda. Asegúrate de reemplazar el fragmento `### TU CÓDIGO AQUÍ ###` con tu propio código antes de ejecutar la siguiente celda.

[10]:
```

```

```
# Asigna a `import_file` el nombre del archivo de texto que contiene el archivo␣
 ↪de registro de seguridad

import_file = "data/login.txt"

# La sentencia `with`
# Utiliza `open()` para importar el archivo de registro de seguridad y␣
 ↪almacenarlo como una cadena

with open(import_file, "r") as file:

  # Utiliza `.read()` para leer el archivo importado y almacenar el resultado␣
 ↪en una variable llamada `text`

  text = file.read()

# Muestra el contenido de `text`

print(text)
```

```
username,ip_address,time,date
tshah,192.168.92.147,15:26:08,2022-05-10
dtanaka,192.168.98.221,9:45:18,2022-05-09
tmitchel,192.168.110.131,14:13:41,2022-05-11
daquino,192.168.168.144,7:02:35,2022-05-08
eraab,192.168.170.243,1:45:14,2022-05-11
jlansky,192.168.238.42,1:07:11,2022-05-11
acook,192.168.52.90,9:56:48,2022-05-10
asundara,192.168.58.217,23:17:52,2022-05-12
jclark,192.168.214.49,20:49:00,2022-05-10
cjackson,192.168.247.153,19:36:42,2022-05-12
jclark,192.168.197.247,14:11:04,2022-05-12
apatel,192.168.46.207,17:39:42,2022-05-10
mabadi,192.168.96.244,10:24:43,2022-05-12
iuduike,192.168.131.147,17:50:00,2022-05-11
abellmas,192.168.60.111,13:37:05,2022-05-10
gesparza,192.168.148.80,6:30:14,2022-05-11
cgriffin,192.168.4.157,23:04:05,2022-05-09
alevitsk,192.168.210.228,8:10:43,2022-05-08
eraab,192.168.24.12,11:29:27,2022-05-11
jsoto,192.168.25.60,5:09:21,2022-05-09
jrafael,192.168.243.140,4:56:27,2022-05-09jrafael,192.168.243.140,4:56:27,2022-0
5-09jrafael,192.168.243.140,4:56:27,2022-05-09jrafael,192.168.243.140,4:56:27,20
22-05-09jrafael,192.168.243.140,4:56:27,2022-05-09jrafael,192.168.243.140,4:56:2
7,2022-05-09jrafael,192.168.243.140,4:56:27,2022-05-09jrafael,192.168.243.140,4:
56:27,2022-05-09jrafael,192.168.243.140,4:56:27,2022-05-09jrafael,192.168.243.14
0,4:56:27,2022-05-09
```

Pista 1

El método `.read()` en Python convierte archivos de texto en cadenas.

Pista 2

El objeto `file` contiene el archivo que deseas leer, así que aplica el método `.read()` a `file`.

Pista 3

Utiliza la función `print()` para mostrar el contenido de `text`.

Tarea 3 La salida en el paso anterior es una cadena grande, que sería difícil de analizar. Para mejorar esto, puedes dividir la cadena que contiene todo el archivo de registro importado en una lista de cadenas, con una cadena por línea.

Utiliza el método `.split()` para realizar esta división y luego muestra el resultado. Asegúrate de reemplazar el fragmento `### TU CÓDIGO AQUÍ ###` con tu propio código antes de ejecutar la siguiente celda.

Ten en cuenta que mostrar `.split()` no cambia lo que se almacena en la variable `text`. La reasignación de variables sería necesaria si quieres almacenar el resultado después de dividir.

[11]:
```
# Asigna a `import_file` el nombre del archivo de texto que contiene el archivo␣
↪de registro de seguridad

import_file = "data/login.txt"

# La sentencia `with`
# Utiliza `open()` para importar el archivo de registro de seguridad y␣
↪almacenarlo como una cadena

with open(import_file, "r") as file:

  # Utiliza `.read()` para leer el archivo importado y almacenar el resultado␣
↪en una variable llamada `text`

  text = file.read()

# Muestra el contenido de `text` dividido en líneas separadas

print(text.split())
```

```
['username,ip_address,time,date', 'tshah,192.168.92.147,15:26:08,2022-05-10',
'dtanaka,192.168.98.221,9:45:18,2022-05-09',
'tmitchel,192.168.110.131,14:13:41,2022-05-11',
'daquino,192.168.168.144,7:02:35,2022-05-08',
'eraab,192.168.170.243,1:45:14,2022-05-11',
'jlansky,192.168.238.42,1:07:11,2022-05-11',
'acook,192.168.52.90,9:56:48,2022-05-10',
'asundara,192.168.58.217,23:17:52,2022-05-12',
'jclark,192.168.214.49,20:49:00,2022-05-10',
```

```
'cjackson,192.168.247.153,19:36:42,2022-05-12',
'jclark,192.168.197.247,14:11:04,2022-05-12',
'apatel,192.168.46.207,17:39:42,2022-05-10',
'mabadi,192.168.96.244,10:24:43,2022-05-12',
'iuduike,192.168.131.147,17:50:00,2022-05-11',
'abellmas,192.168.60.111,13:37:05,2022-05-10',
'gesparza,192.168.148.80,6:30:14,2022-05-11',
'cgriffin,192.168.4.157,23:04:05,2022-05-09',
'alevitsk,192.168.210.228,8:10:43,2022-05-08',
'eraab,192.168.24.12,11:29:27,2022-05-11',
'jsoto,192.168.25.60,5:09:21,2022-05-09', 'jrafael,192.168.243.140,4:56:27,2022-
05-09jrafael,192.168.243.140,4:56:27,2022-05-09jrafael,192.168.243.140,4:56:27,2
022-05-09jrafael,192.168.243.140,4:56:27,2022-05-09jrafael,192.168.243.140,4:56:
27,2022-05-09jrafael,192.168.243.140,4:56:27,2022-05-09jrafael,192.168.243.140,4
:56:27,2022-05-09jrafael,192.168.243.140,4:56:27,2022-05-09jrafael,192.168.243.1
40,4:56:27,2022-05-09jrafael,192.168.243.140,4:56:27,2022-05-09']
```

Pista 1

El método `.split()` en Python convierte una cadena en una lista. Puede tomar un carácter separador que especifique en qué carácter dividir. Si no se especifica un carácter, se dividirá en espacios en blanco de forma predeterminada. Este valor predeterminado funcionará bien para tu tarea, ya que el archivo de registro contiene espacios en blanco entre cada línea del registro.

Ten en cuenta que el espacio en blanco incluye cualquier espacio entre el texto de la misma línea y el espacio entre una línea y la siguiente.

Pista 2

Utiliza el método `.split()` para convertir el `text` en una lista, donde cada elemento en ella representa una línea en el archivo de registro.

Coloca esto entre los paréntesis en la llamada a la función `print()`.

Pregunta 1 ¿Qué observas en la salida antes y después de usar el método `.split()`?

Antes de usar el método `.split()`, la salida es una cadena larga que contiene todas las líneas del archivo de registro. Después de usar el método `.split()`, la salida es una lista de cadenas, donde cada cadena corresponde a una línea del archivo de registro.

1.3 Tarea 4

Falta una entrada en el archivo de registro. Deberás tenerla en cuenta agregándola al archivo de registro. Se te proporciona la entrada que falta, almacenada en una variable llamada `missing_entry`. Utiliza el método `.write()` y el parámetro `"a"` en la función `open()`. Asegúrate de reemplazar cada `### TU CÓDIGO AQUÍ ###` con tu propio código antes de ejecutar la siguiente celda. Ten en cuenta que la celda de código contendrá una sentencia `with` que escribe en un archivo, pero no muestra información en la pantalla, por lo que su ejecución no producirá ninguna salida.

[12]:
```
# Asigna a `import_file` el nombre del archivo de texto que contiene el archivo␣
 ↪de registro de seguridad

import_file = "data/login.txt"

# Asigna `missing entry` a un registro que no se incluyó en el archivo de␣
 ↪registro

missing_entry = "jrafael,192.168.243.140,4:56:27,2022-05-09"

# Utiliza `open()` para importar el archivo de registro de seguridad y␣
 ↪almacenarlo como una cadena
# Pasa la letra "a" como segundo parámetro cuando quieras indicar que el␣
 ↪archivo se está abriendo con fines de adición

with open(import_file, "a") as file:

    # Utiliza `.write()` para agregar `missing_entry` al archivo de registro

    file.write(missing_entry)
```

Pista 1

La función `open()` en Python te permite abrir un archivo.

Como primer parámetro, toma el nombre del archivo (o una variable que contiene el nombre del archivo). Como segundo parámetro, toma una cadena que indica cómo se debe gestionar el archivo.

Pasa la letra `"a"` como segundo parámetro cuando quieras agregar contenido al archivo.

Pista 2

Llama al método `.write()` en el archivo de registro y pasa `missing_entry`. Esto agregará `missing_entry` al archivo de registro.

Pista 3

Llama a `file.write ()` y pasa `missing_entry`. Esto agregará `missing_entry` al archivo de registro.

Tarea 5 Ahora agregarás más código para leer el contenido actualizado del archivo.

En primer lugar, completa la siguiente sentencia `with` para leer el contenido de `import_file`. Especifica `"r"` como segundo parámetro al llamar a la función `open()`. Utiliza el método `.read()` al almacenar el archivo en una nueva variable llamada `text`. Muestra el contenido del archivo almacenado en `text` y ejecuta la celda para observar el resultado. Asegúrate de reemplazar cada `### TU CÓDIGO AQUÍ ###` con tu propio código antes de ejecutar la siguiente celda.

[13]:
```
# Asigna a `import_file` el nombre del archivo de texto que contiene el archivo␣
 ↪de registro de seguridad

```

```
import_file = "data/login.txt"

# Asigna `missing entry` a un registro que no se incluyó en el archivo de␣
 ↪registro

missing_entry = "jrafael,192.168.243.140,4:56:27,2022-05-09"

# Utiliza `open()` para importar el archivo de registro de seguridad y␣
 ↪almacenarlo como una cadena
# Pasa la letra "a" como segundo parámetro cuando quieras indicar que el␣
 ↪archivo se está abriendo con fines de adición

with open(import_file, "a") as file:

    # Utiliza `.write()` para agregar `missing_entry` al archivo de registro

    file.write(missing_entry)

# Utiliza `open()` con el parámetro "r" para abrir el archivo de registro de␣
 ↪seguridad con fines de lectura

with open(import_file, "r") as file:

    # Utiliza `.read()` para leer el contenido del archivo de registro y␣
 ↪almacenarlo en una variable llamada `text`

    text = file.read()

# Muestra el contenido de `text`

print(text)
```

```
username,ip_address,time,date
tshah,192.168.92.147,15:26:08,2022-05-10
dtanaka,192.168.98.221,9:45:18,2022-05-09
tmitchel,192.168.110.131,14:13:41,2022-05-11
daquino,192.168.168.144,7:02:35,2022-05-08
eraab,192.168.170.243,1:45:14,2022-05-11
jlansky,192.168.238.42,1:07:11,2022-05-11
acook,192.168.52.90,9:56:48,2022-05-10
asundara,192.168.58.217,23:17:52,2022-05-12
jclark,192.168.214.49,20:49:00,2022-05-10
cjackson,192.168.247.153,19:36:42,2022-05-12
jclark,192.168.197.247,14:11:04,2022-05-12
apatel,192.168.46.207,17:39:42,2022-05-10
mabadi,192.168.96.244,10:24:43,2022-05-12
iuduike,192.168.131.147,17:50:00,2022-05-11
```

```
abellmas,192.168.60.111,13:37:05,2022-05-10
gesparza,192.168.148.80,6:30:14,2022-05-11
cgriffin,192.168.4.157,23:04:05,2022-05-09
alevitsk,192.168.210.228,8:10:43,2022-05-08
eraab,192.168.24.12,11:29:27,2022-05-11
jsoto,192.168.25.60,5:09:21,2022-05-09
jrafael,192.168.243.140,4:56:27,2022-05-09jrafael,192.168.243.140,4:56:27,2022-0
5-09jrafael,192.168.243.140,4:56:27,2022-05-09jrafael,192.168.243.140,4:56:27,20
22-05-09jrafael,192.168.243.140,4:56:27,2022-05-09jrafael,192.168.243.140,4:56:2
7,2022-05-09jrafael,192.168.243.140,4:56:27,2022-05-09jrafael,192.168.243.140,4:
56:27,2022-05-09jrafael,192.168.243.140,4:56:27,2022-05-09jrafael,192.168.243.14
0,4:56:27,2022-05-09jrafael,192.168.243.140,4:56:27,2022-05-09jrafael,192.168.24
3.140,4:56:27,2022-05-09
```

Pista 1

La función `open()` en Python te permite abrir un archivo.

Como primer parámetro, toma el nombre del archivo (o una variable que contiene el nombre del archivo). Como segundo parámetro, toma una cadena que indica cómo se debe gestionar el archivo.

Pasa la letra `"r"` como segundo parámetro al abrir un archivo con el propósito de leer su contenido.

Pista 2

Llama al método `.read()` en el archivo de registro para leer su contenido. Asigna el resultado a la variable `text`.

Pista 3

Llama a `file.read()`. Coloca esto a la derecha del operador `=` para asignar la salida a la variable `text`.

Pregunta 2 ¿Qué observas en la posición de la entrada que se agregó al archivo de registro?

Se agregó la entrada adicional al final del archivo de registro, por lo que aparece en la última línea de la salida.

1.4 Tarea 6

La siguiente tarea de la que eres responsable es crear un archivo de texto. Este archivo de texto debe incluir una lista de direcciones IP a las que se les permite acceder a información restringida. Documentar esto en un archivo de texto te ayudará a comunicar tus hallazgos a tu equipo de seguridad.

Comienza por crear una variable llamada `import_file` que almacene el nombre del archivo, que debería ser `"allow_list.txt"`.

También se te proporciona una variable llamada `ip_addresses` que almacena una cadena que contiene las direcciones IP permitidas.

Ejecuta el código para mostrar las dos variables y explorar lo que contienen. Asegúrate de reemplazar `### TU CÓDIGO AQUÍ ###` con tu propio código antes de ejecutar la siguiente celda.

[14]:
```
# Asigna a `import_file` el nombre del archivo de texto que deseas crear

import_file = "data/allow_list.txt"

# Asigna a `ip_addresses` una lista de direcciones IP a las que se les permite␣
 ↪acceder a la información restringida

ip_addresses = "192.168.218.160 192.168.97.225 192.168.145.158 192.168.108.13␣
 ↪192.168.60.153 192.168.96.200 192.168.247.153 192.168.3.252 192.168.116.187␣
 ↪192.168.15.110 192.168.39.246"

# Muestra `import_file`

print(import_file)

# Muestra `ip_addresses`

print(ip_addresses)
```

```
data/allow_list.txt
192.168.218.160 192.168.97.225 192.168.145.158 192.168.108.13 192.168.60.153
192.168.96.200 192.168.247.153 192.168.3.252 192.168.116.187 192.168.15.110
192.168.39.246
```

Pista 1

Ten en cuenta que el nombre del archivo de texto que deseas crear debe ser `"allow_list.txt"`. Asegúrate de incluir la extensión de archivo `.txt`, que especifica el formato de archivo.

1.5 Tarea 7

El siguiente objetivo es crear una sentencia `with` para escribir las direcciones IP en el archivo de texto que creaste en el paso anterior.

Primero abrirás el archivo con el parámetro `"w"`. Luego, escribirás las direcciones IP en el archivo. Asegúrate de reemplazar cada `### TU CÓDIGO AQUÍ ###` con tu propio código antes de ejecutar la siguiente celda. Ten en cuenta que la celda de código contendrá una sentencia `with` que escribe en un archivo, pero no muestra información en la pantalla, por lo que su ejecución no producirá ninguna salida.

[15]:
```
# Asigna a `import_file` el nombre del archivo de texto que deseas crear

import_file = "data/allow_list.txt"
```

```
# Asigna a `ip_addresses` una lista de direcciones IP a las que se les permite␣
 ↪acceder a la información restringida

ip_addresses = "192.168.218.160 192.168.97.225 192.168.145.158 192.168.108.13␣
 ↪192.168.60.153 192.168.96.200 192.168.247.153 192.168.3.252 192.168.116.187␣
 ↪192.168.15.110 192.168.39.246"

# Crea una sentencia `with` para escribir en el archivo de texto

with open(import_file, "w") as file:

  # Escribe `ip_addresses` en el archivo de texto

  file.write(ip_addresses)
```

Pista 1

La función `open()` en Python te permite abrir un archivo.

Como primer parámetro, toma el nombre del archivo (o una variable que contiene el nombre del archivo). Como segundo parámetro, toma una cadena que indica cómo se debe gestionar el archivo.

Pasa la letra `"w"` como segundo parámetro al abrir un archivo con el propósito de escribir en este.

Pista 2

Llama al método `.write()` en el archivo de texto para escribir en este.

Pista 3

Llama al método `file.write()` y pasa a la variable `ip_addresses` para escribir el contenido de esa variable en el archivo de texto.

1.6 Tarea 8

En este último paso, completarás el código que has estado escribiendo hasta este punto. Agregarás código para leer el archivo que contiene las direcciones IP.

Completa una sentencia `with` que lea el archivo de texto y lo almacene en una nueva variable llamada `text`.

Después, muestra el contenido de `text` y ejecuta la celda para explorar el resultado. Asegúrate de reemplazar cada `### TU CÓDIGO AQUÍ ###` con tu propio código antes de ejecutar la siguiente celda.

[16]:
```
# Asigna a `import_file` el nombre del archivo de texto que deseas crear

import_file = "data/allow_list.txt"

# Asigna a `ip_addresses` una lista de direcciones IP a las que se les permite␣
 ↪acceder a la información restringida
```

```
ip_addresses = "192.168.218.160 192.168.97.225 192.168.145.158 192.168.108.13␣
↪192.168.60.153 192.168.96.200 192.168.247.153 192.168.3.252 192.168.116.187␣
↪192.168.15.110 192.168.39.246"

# Crea una sentencia `with` para escribir en el archivo de texto

with open(import_file, "w") as file:

  # Escribe `ip_addresses` en el archivo de texto

    file.write(ip_addresses)

# Crea una sentencia `with` para leer el archivo de texto

with open(import_file, "r") as file:

    # Lee el archivo y almacena el resultado en una variable llamada `text`

    text = file.read()

# Muestra el contenido de `text`

print(text)
```

```
192.168.218.160 192.168.97.225 192.168.145.158 192.168.108.13 192.168.60.153
192.168.96.200 192.168.247.153 192.168.3.252 192.168.116.187 192.168.15.110
192.168.39.246
```

Pista 1

La función `open()` en Python te permite abrir un archivo.

Toma el nombre del archivo como primer parámetro y una cadena que indica cómo se debe gestionar el archivo como segundo parámetro.

Pasa la letra `"r"` como segundo parámetro al abrir un archivo con el propósito de leer su contenido.

Pista 2

Llama al método `.read()` en el archivo de texto para leerlo.

Pista 3

Llama a `file.read()`. Coloca esto a la derecha del operador `=` para asignar la salida a la variable `text`.

1.7 Conclusión

¿Qué conclusiones clave obtuviste de este laboratorio?

— Python tiene funciones y sintaxis que te ayudan a importar y analizar archivos de texto. — La sentencia `with` te permite gestionar archivos de manera eficiente. — La función `open()` te permite importar o abrir un archivo. Toma el nombre del archivo como primer parámetro y una cadena que indica el propósito de abrir el archivo como segundo parámetro. — Especifica `"r"` como segundo parámetro si quieres abrir el archivo con fines de lectura. — Especifica `"a"` como segundo parámetro si quieres abrir el archivo con fines de adición. — Especifica `"w"` como segundo parámetro si quieres abrir el archivo con fines de escritura. — El método `.read()` te permite leer un archivo. — El método `.write()` te permite agregar datos o escribir en un archivo. — El método `.split()` en Python te permite convertir una cadena en una lista.