LAB-011-Exemplar_Use regular expressions to find patterns

September 20, 2024

1 Modelo: Uso de expresiones regulares para encontrar patrones

Introducción

Los analistas de seguridad a menudo analizan los archivos de registro, incluidos los que contienen información sobre los intentos de inicio de sesión. Por ejemplo, como analista, puedes marcar las direcciones IP que se relacionan con intentos inusuales para iniciar sesión en el sistema.

Otra área de enfoque en ciberseguridad es la detección de dispositivos que requieren actualizaciones. Las actualizaciones de software ayudan a prevenir los problemas de seguridad debido a las vulnerabilidades.

El uso de expresiones regulares en Python puede ayudar a automatizar los procesos involucrados en ambas áreas de la ciberseguridad. Los patrones y funciones de expresión regular se pueden utilizar para extraer de manera eficiente información importante de cadenas y archivos.

En este laboratorio, escribirás expresiones regulares para extraer información como ID del dispositivo o direcciones IP.

Consejos para completar este laboratorio

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

— `#### YOUR CODE HERE ###` 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 y tus principales tareas serán las siguientes: — Extraer las ID de los dispositivos que contienen ciertos caracteres de un registro; estos se corresponden con un determinado sistema operativo que requiere una actualización. — Extraer todas las direcciones IP de un registro y luego compararlas con las que están marcadas en una lista.

Tarea 1 A fin de trabajar con expresiones regulares en Python, comienza por importar la biblioteca `re`. Esta biblioteca contiene muchas funciones que te ayudarán a trabajar con expresiones regulares. Al ejecutar la siguiente celda de código, la biblioteca estará disponible a través del resto del cuaderno.

[1]:
```
# Importa la biblioteca `re` en Python

import re
```

1.2 Tarea 2

En tu trabajo como analista de ciberseguridad, eres responsable de actualizar los dispositivos. Una ID de dispositivo que comienza con los caracteres `"r15"` indica que el dispositivo tiene un sistema operativo determinado que debe actualizarse.

Se te da un registro de las ID de dispositivo, almacenadas en una variable llamada `devices`. Tu objetivo final es extraer las ID de los dispositivos que comienzan con los caracteres `"r15"`. Por ahora, muestra el contenido de toda la cadena para examinar lo que contiene. Asegúrate de reemplazar el fragmento `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda.

[2]:
```
# Asigna a `devices` una cadena que contiene las ID de dispositivo, cada una de␣
 ↪ellas representada por caracteres alfanuméricos

devices = "r262c36 67bv8fy 41j1u2e r151dm4 1270t3o 42dr56i r15xk9h 2j33krk␣
 ↪253be78 ac742a1 r15u9q5 zh86b2l ii286fq 9x482kt 6oa6m6u x3463ac i4l56nq␣
 ↪g07h55q 081qc9t r159r1u"

# Muestra el contenido de `devices`

print(devices)
```

```
r262c36 67bv8fy 41j1u2e r151dm4 1270t3o 42dr56i r15xk9h 2j33krk 253be78 ac742a1
r15u9q5 zh86b2l ii286fq 9x482kt 6oa6m6u x3463ac i4l56nq g07h55q 081qc9t r159r1u
```

Pista 1

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

Tarea 3 En esta tarea, escribirás un patrón para encontrar dispositivos que comiencen con la combinación de caracteres de `"r15"`.

Utiliza los símbolos de expresión regular `\w` y `+` para crear el patrón y guárdalo como una cadena en una variable llamada `target_pattern`.

Asegúrate de reemplazar el fragmento `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda. Ten en cuenta que la celda de código contendrá solo asignaciones de variables, por lo que su ejecución no producirá ningún resultado.

[3]:
```
# Asigna a `devices` una cadena que contiene las ID de dispositivo, cada una de␣
 ↪ellas representada por caracteres alfanuméricos
```

```
devices = "r262c36 67bv8fy 41j1u2e r151dm4 1270t3o 42dr56i r15xk9h 2j33krk␣
 ↪253be78 ac742a1 r15u9q5 zh86b2l ii286fq 9x482kt 6oa6m6u x3463ac i4l56nq␣
 ↪g07h55q 081qc9t r159r1u"

# Asigna `target_pattern` a un patrón de expresión regular para encontrar las␣
 ↪ID de dispositivo que comiencen con "r15"

target_pattern = "r15\w+"
```

Pista 1

El símbolo de expresión regular `\w` coincide con cualquier carácter alfanumérico.

El símbolo de expresión regular `+` coincide con una o más apariciones del carácter anterior en el patrón.

Pista 2

La combinación de los símbolos de expresión regular `\w` y `+` da como resultado `\w+`, que coincide con una cadena alfanumérica de cualquier longitud.

Pista 3

Dado que `target_pattern` debe coincidir con una cadena alfanumérica que comience con `"r15"`, asegúrate de que el patrón comience con `r15`, seguido de `\w+`.

Pregunta 1 ¿Qué patrón de expresión regular utilizaste? Para cada componente del patrón, ¿qué pasaría si faltara?

El patrón de expresión regular utilizado fue `"r15\w+"`. Sin la `r15`, el patrón no coincidirá con las ID de dispositivo que comienzan con los caracteres `"r15"`. Sin la `\w`, el patrón no coincidirá con los caracteres alfanuméricos que siguen a `"r15"`. Sin el `+`, el patrón coincidirá solo con el primer carácter alfanumérico que se produce después de `"r15"`.

1.3 Tarea 4

Utiliza la función `findall()` de la biblioteca `re` para encontrar las ID de dispositivo con los que coincide el `target_pattern`. Asegúrate de reemplazar el fragmento `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda.

[4]:
```
# Asigna a `devices` una cadena que contiene las ID de dispositivo, cada una de␣
 ↪ellas representada por caracteres alfanuméricos

devices = "r262c36 67bv8fy 41j1u2e r151dm4 1270t3o 42dr56i r15xk9h 2j33krk␣
 ↪253be78 ac742a1 r15u9q5 zh86b2l ii286fq 9x482kt 6oa6m6u x3463ac i4l56nq␣
 ↪g07h55q 081qc9t r159r1u"
```

```
# Asigna `target_pattern` a un patrón de expresión regular para encontrar las␣
 ↪ID de dispositivo que comiencen con "r15"

target_pattern = "r15\w+"

# Utiliza `re.findall()` para encontrar las ID de dispositivo que comienzan con␣
 ↪"r15" y muestra los resultados

print(re.findall(target_pattern, devices))
```

```
['r151dm4', 'r15xk9h', 'r15u9q5', 'r159r1u']
```

Pista 1

La función `findall()` de la biblioteca `re` toma una expresión regular, seguida de una cadena. La función aplica la expresión regular a la cadena y devuelve una lista de coincidencias.

Pista 2

Al llamar a la función `re.findall()`, pasa la variable `target_pattern` como primer argumento y la variable `devices` como segundo. Esto asegurará que `target_pattern` se aplique a la cadena almacenada en `devices`.

Tarea 5 Ahora, la siguiente tarea de la que eres responsable es analizar un archivo de registro de seguridad de red y determinar qué direcciones IP se denunciaron por actividad inusual.

Se te proporciona el archivo de registro como una cadena almacenada en una variable llamada `log_file`. Hay algunas direcciones IP no válidas en el archivo de registro debido a problemas en la recopilación de datos. Tu objetivo final es utilizar expresiones regulares para extraer las direcciones IP válidas de la cadena.

Comienza por mostrar el contenido del `log_file` para examinar sus detalles. Asegúrate de reemplazar el fragmento `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda.

[5]:
```
# Asigna a `log_file` una cadena que contiene el nombre de usuario, la fecha,␣
 ↪la hora de inicio de sesión y la dirección IP para una serie de intentos de␣
 ↪inicio de sesión

log_file = "eraab 2022-05-10 6:03:41 192.168.152.148 \niuduike 2022-05-09 6:46:
 ↪40 192.168.22.115 \nsmartell 2022-05-09 19:30:32 192.168.190.178 \narutley␣
 ↪2022-05-12 17:00:59 1923.1689.3.24 \nrjensen 2022-05-11 0:59:26 192.168.213.
 ↪128 \naestrada 2022-05-09 19:28:12 1924.1680.27.57 \nasundara 2022-05-11 18:
 ↪38:07 192.168.96.200 \ndkot 2022-05-12 10:52:00 1921.168.1283.75 \nabernard␣
 ↪2022-05-12 23:38:46 19245.168.2345.49 \ncjackson 2022-05-12 19:36:42 192.168.
 ↪247.153 \njclark 2022-05-10 10:48:02 192.168.174.117 \nalevitsk 2022-05-08␣
 ↪12:09:10 192.16874.1390.176 \njrafael 2022-05-10 22:40:01 192.168.148.115␣
 ↪\nyappiah 2022-05-12 10:37:22 192.168.103.10654 \ndaquino 2022-05-08 7:02:35␣
 ↪192.168.168.144"

# Muestra el contenido de `log_file`
```

```
print(log_file)
```

```
eraab 2022-05-10 6:03:41 192.168.152.148
iuduike 2022-05-09 6:46:40 192.168.22.115
smartell 2022-05-09 19:30:32 192.168.190.178
arutley 2022-05-12 17:00:59 1923.1689.3.24
rjensen 2022-05-11 0:59:26 192.168.213.128
aestrada 2022-05-09 19:28:12 1924.1680.27.57
asundara 2022-05-11 18:38:07 192.168.96.200
dkot 2022-05-12 10:52:00 1921.168.1283.75
abernard 2022-05-12 23:38:46 19245.168.2345.49
cjackson 2022-05-12 19:36:42 192.168.247.153
jclark 2022-05-10 10:48:02 192.168.174.117
alevitsk 2022-05-08 12:09:10 192.16874.1390.176
jrafael 2022-05-10 22:40:01 192.168.148.115
yappiah 2022-05-12 10:37:22 192.168.103.10654
daquino 2022-05-08 7:02:35 192.168.168.144
```

Pista 1

Utiliza la función `print()` para mostrar el contenido del `log_file`.

1.4 Tarea 6

En esta tarea, crearás un patrón de expresión regular que puedes utilizar más adelante para extraer direcciones IP en forma de xxx.xxx.xxx.xxx.xxx. En otras palabras, extraerás todas las direcciones IP que contengan cuatro segmentos de tres dígitos que estén separados por puntos.

Escribe un patrón de expresión regular que coincida con estas direcciones IP y guárdalo en una variable llamada `pattern`. Utiliza los símbolos de expresión regular `\d` y `\.` en tu patrón. Ten en cuenta que el símbolo `\d` coincide con los dígitos, en otras palabras, cualquier número entero entre 0 y 9. Asegúrate de reemplazar el fragmento `### YOUR CODE HERE ###` con tu propio código. Dado que solo crearás el patrón aquí, no habrá ningún resultado cuando ejecutes esta celda.

[6]:
```
# Asigna a `log_file` una cadena que contiene el nombre de usuario, la fecha,␣
↪la hora de inicio de sesión y la dirección IP para una serie de intentos de␣
↪inicio de sesión
```

```
log_file = "eraab 2022-05-10 6:03:41 192.168.152.148 \niuduike 2022-05-09 6:46:
 ↪40 192.168.22.115 \nsmartell 2022-05-09 19:30:32 192.168.190.178 \narutley␣
 ↪2022-05-12 17:00:59 1923.1689.3.24 \nrjensen 2022-05-11 0:59:26 192.168.213.
 ↪128 \naestrada 2022-05-09 19:28:12 1924.1680.27.57 \nasundara 2022-05-11 18:
 ↪38:07 192.168.96.200 \ndkot 2022-05-12 10:52:00 1921.168.1283.75 \nabernard␣
 ↪2022-05-12 23:38:46 19245.168.2345.49 \ncjackson 2022-05-12 19:36:42 192.168.
 ↪247.153 \njclark 2022-05-10 10:48:02 192.168.174.117 \nalevitsk 2022-05-08␣
 ↪12:09:10 192.16874.1390.176 \njrafael 2022-05-10 22:40:01 192.168.148.115␣
 ↪\nyappiah 2022-05-12 10:37:22 192.168.103.10654 \ndaquino 2022-05-08 7:02:35␣
 ↪192.168.168.144"

# Asigna a `pattern` un patrón de expresión regular que coincida con las␣
 ↪direcciones IP de la forma xxx.xxx.xxx.xxx
pattern = "\d\d\d\.\d\d\d\.\d\d\d\.\d\d\d"
```

Pista 1

El símbolo `\.` coincide con los puntos.

Pista 2

Recuerda que esta tarea se centra en las direcciones IP en forma de xxx.xxx.xxx.xxx, donde cada x es un dígito. En otras palabras, estas direcciones tienen el siguiente formato: tres dígitos seguidos por un punto, tres dígitos seguidos por un punto, tres dígitos seguidos por un punto, tres dígitos.

Para crear un patrón que coincida con las direcciones IP de esta forma, utiliza el símbolo `\d` para cada dígito y el `\.` para cada punto que debería estar en la dirección IP.

Pista 3

Puedes utilizar la expresión regular `\d\d\d\d\.` para que coincida con un segmento de tres dígitos seguido de un punto.

1.5 Tarea 7

En esta tarea, utilizarás la función `re.findall()` en el patrón de expresión regular almacenado en la variable `pattern` y el `log_file` proporcionado para extraer las direcciones IP correspondientes. A continuación, ejecuta la celda y toma nota del resultado que genera. Asegúrate de reemplazar el fragmento `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda.

[7]:
```
# Asigna a `log_file` una cadena que contiene el nombre de usuario, la fecha,␣
 ↪la hora de inicio de sesión y la dirección IP para una serie de intentos de␣
 ↪inicio de sesión
```

```
log_file = "eraab 2022-05-10 6:03:41 192.168.152.148 \niuduike 2022-05-09 6:46:
 ↪40 192.168.22.115 \nsmartell 2022-05-09 19:30:32 192.168.190.178 \narutley␣
 ↪2022-05-12 17:00:59 1923.1689.3.24 \nrjensen 2022-05-11 0:59:26 192.168.213.
 ↪128 \naestrada 2022-05-09 19:28:12 1924.1680.27.57 \nasundara 2022-05-11 18:
 ↪38:07 192.168.96.200 \ndkot 2022-05-12 10:52:00 1921.168.1283.75 \nabernard␣
 ↪2022-05-12 23:38:46 19245.168.2345.49 \ncjackson 2022-05-12 19:36:42 192.168.
 ↪247.153 \njclark 2022-05-10 10:48:02 192.168.174.117 \nalevitsk 2022-05-08␣
 ↪12:09:10 192.16874.1390.176 \njrafael 2022-05-10 22:40:01 192.168.148.115␣
 ↪\nyappiah 2022-05-12 10:37:22 192.168.103.10654 \ndaquino 2022-05-08 7:02:35␣
 ↪192.168.168.144"

# Asigna a `pattern` un patrón de expresión regular que coincida con las␣
 ↪direcciones IP de la forma xxx.xxx.xxx.xxx

pattern = "\d\d\d\.\d\d\d\.\d\d\d\.\d\d\d"

# Utiliza la función `re.findall()` en `pattern` y `log_file` para extraer las␣
 ↪direcciones IP de la forma xxx.xxx.xxx.xxx y mostrar los resultados

print(re.findall(pattern, log_file))
```

```
['192.168.152.148', '192.168.190.178', '192.168.213.128', '192.168.247.153',
'192.168.174.117', '192.168.148.115', '192.168.103.106', '192.168.168.144']
```

Pista 1

La función `re.findall()` toma una expresión regular, seguida de una cadena. La función aplica la expresión regular a la cadena y devuelve una lista de coincidencias.

Pista 2

Al llamar a la función `re.findall()`, pasa la variable `pattern` como primer argumento y la variable `log_file` como segundo. Esto asegurará que `pattern` se aplique a la cadena almacenada en `log_file`.

Pregunta 2 ¿Cuáles son algunos ejemplos de direcciones IP que se extrajeron? ¿Cuáles son algunos ejemplos de direcciones IP que no se extrajeron? ¿Parecen algunas que no se extrajeron ser direcciones IP válidas?

Ejemplos de direcciones IP que se extrajeron incluyen `"192.168.152.148"` y `"192.168.190.178"`. Ejemplos de direcciones IP que no se extrajeron incluyen `"192.168.22.115"` y `"1923.1689.3.24"`. Las direcciones IP que tienen menos de tres dígitos por segmento, como `"192.168.22.115"` (que tiene dos dígitos en el tercer segmento y tres dígitos en cada uno de los otros segmentos), son direcciones IP válidas pero no se extrajeron.

1.6 Tarea 8

Hay algunas direcciones IP válidas en el `log_file` que aún no extrajiste. Esto se debe a que cada segmento de dígitos en una dirección IP válida puede tener entre uno y tres dígitos.

Ajusta la expresión regular en el `pattern` para permitir la variación en el número de dígitos en cada segmento. Puedes hacerlo mediante el símbolo `+` después de `\d`. Luego, utiliza el `pattern` actualizado para extraer las direcciones IP restantes. A continuación, ejecuta la celda para analizar los resultados. Asegúrate de reemplazar el fragmento `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda.

[8]:
```
# Asigna a `log_file` una cadena que contiene el nombre de usuario, la fecha, la hora de inicio de sesión y la dirección IP para una serie de intentos de inicio de sesión

log_file = "eraab 2022-05-10 6:03:41 192.168.152.148 \niuduike 2022-05-09 6:46:40 192.168.22.115 \nsmartell 2022-05-09 19:30:32 192.168.190.178 \narutley 2022-05-12 17:00:59 1923.1689.3.24 \nrjensen 2022-05-11 0:59:26 192.168.213.128 \naestrada 2022-05-09 19:28:12 1924.1680.27.57 \nasundara 2022-05-11 18:38:07 192.168.96.200 \ndkot 2022-05-12 10:52:00 1921.168.1283.75 \nabernard 2022-05-12 23:38:46 19245.168.2345.49 \ncjackson 2022-05-12 19:36:42 192.168.247.153 \njclark 2022-05-10 10:48:02 192.168.174.117 \nalevitsk 2022-05-08 12:09:10 192.16874.1390.176 \njrafael 2022-05-10 22:40:01 192.168.148.115 \nyappiah 2022-05-12 10:37:22 192.168.103.10654 \ndaquino 2022-05-08 7:02:35 192.168.168.144"

# Actualiza el `pattern` a un patrón de expresión regular que coincida con las direcciones IP con cualquier variación en el número de dígitos por segmento

pattern = "\d+\.\d+\.\d+\.\d+"

# Utiliza la función `re.findall()` en `pattern` y `log_file` para extraer las direcciones IP del formulario actualizado especificado anteriormente y muestra los resultados

print(re.findall(pattern, log_file))
```

```
['192.168.152.148', '192.168.22.115', '192.168.190.178', '1923.1689.3.24',
'192.168.213.128', '1924.1680.27.57', '192.168.96.200', '1921.168.1283.75',
'19245.168.2345.49', '192.168.247.153', '192.168.174.117', '192.16874.1390.176',
'192.168.148.115', '192.168.103.10654', '192.168.168.144']
```

Pista 1

El símbolo de expresión regular `+` representa una o más apariciones de un carácter específico.

El símbolo de expresión regular `\d` coincide con los dígitos, en otras palabras, cualquier número entero entre 0 y 9.

Pista 2

Colocar `+` después de `\d` da como resultado `\d+`, que coincidirá con uno o más dígitos.

Pregunta 3 ¿Qué se extrae aquí? ¿Tienen todas las direcciones IP extraídas entre uno y tres dígitos en cada segmento?

Ahora, las direcciones IP extraídas incluyen aquellas con exactamente tres dígitos por segmento (como `"192.168.152.148"`), aquellas con menos de tres dígitos por segmento (como `"192.168.22.115"`) y aquellas con más de tres dígitos por segmento (como `"1923.1689.3.24"`). No todas las direcciones IP extraídas tienen entre uno y tres dígitos en cada segmento.

1.7 Tarea 9

Ten en cuenta que todas las direcciones IP ahora se extraen, pero también incluyen direcciones IP no válidas con más de tres dígitos por segmento.

En esta tarea, actualizarás el `pattern` utilizando corchetes en lugar del símbolo `+`. En expresiones regulares, los corchetes se pueden usar para representar un número exacto de repeticiones entre dos números. Por ejemplo, `{2,4}` en una expresión regular significa entre 2 y 4 ocurrencias de algo. Al aplicar esto a un ejemplo, `\w{2,4}` coincidiría con dos, tres o cuatro caracteres alfanuméricos. Después, llamarás a la función `re.findall()` en el `pattern` actualizado y al `log_file` y almacenarás el resultado en una variable llamada `valid_ip_addresses`.

A continuación, muestra el contenido de `valid_ip_addresses` y ejecuta la celda para analizar los resultados. Asegúrate de reemplazar cada `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda.

[9]:
```
# Asigna a `log_file` una cadena que contiene el nombre de usuario, la fecha,␣
 ↪la hora de inicio de sesión y la dirección IP para una serie de intentos de␣
 ↪inicio de sesión

log_file = "eraab 2022-05-10 6:03:41 192.168.152.148 \niuduike 2022-05-09 6:46:
 ↪40 192.168.22.115 \nsmartell 2022-05-09 19:30:32 192.168.190.178 \narutley␣
 ↪2022-05-12 17:00:59 1923.1689.3.24 \nrjensen 2022-05-11 0:59:26 192.168.213.
 ↪128 \naestrada 2022-05-09 19:28:12 1924.1680.27.57 \nasundara 2022-05-11 18:
 ↪38:07 192.168.96.200 \ndkot 2022-05-12 10:52:00 1921.168.1283.75 \nabernard␣
 ↪2022-05-12 23:38:46 19245.168.2345.49 \ncjackson 2022-05-12 19:36:42 192.168.
 ↪247.153 \njclark 2022-05-10 10:48:02 192.168.174.117 \nalevitsk 2022-05-08␣
 ↪12:09:10 192.16874.1390.176 \njrafael 2022-05-10 22:40:01 192.168.148.115␣
 ↪\nyappiah 2022-05-12 10:37:22 192.168.103.10654 \ndaquino 2022-05-08 7:02:35␣
 ↪192.168.168.144"

# Asigna `pattern` a una expresión regular que coincida con todas las␣
 ↪direcciones IP válidas y solo aquellas

pattern = "\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}"

# Utiliza `re.findall()` en `pattern` y `log_file` y asigna␣
 ↪`valid_ip_addresses` al resultado
```

```
valid_ip_addresses = re.findall(pattern, log_file)

# Muestra el contenido de `valid_ip_addresses`

print(valid_ip_addresses)
```

```
['192.168.152.148', '192.168.22.115', '192.168.190.178', '192.168.213.128',
'192.168.96.200', '192.168.247.153', '192.168.174.117', '192.168.148.115',
'192.168.103.106', '192.168.168.144']
```

Pista 1

Recuerda que los corchetes en las expresiones regulares coinciden con un número de repeticiones entre dos números especificados.

Para crear un patrón de expresión regular que coincida con cualquier lugar entre uno y tres dígitos, utiliza `\d{1,3}`.

Pista 2

Recordemos que una dirección IP válida consta de cuatro segmentos de tres dígitos cada uno, separados por puntos.

Para representar un segmento de tres dígitos seguido de un punto, utiliza `\d{1,3}\.` en el patrón de expresión regular que construyas.

Pregunta 4 ¿Qué observas en las direcciones IP extraídas aquí en comparación con las extraídas en las dos tareas anteriores?

Aquí, todas las direcciones IP extraídas tienen entre uno y tres dígitos por segmento. Recuerda que en la Tarea 7, solo se extrajeron direcciones IP con exactamente tres dígitos por segmento. Y en la Tarea 8, también se extrajeron direcciones IP con más de tres dígitos por segmento.

1.8 Tarea 10

Ahora, se extrajeron todas las direcciones IP válidas. El siguiente paso es identificar las direcciones IP denunciadas.

Se te proporciona una lista de direcciones IP que se denunciaron previamente por actividad inusual, almacenada en una variable llamada `flagged_addresses`. Cuando se encuentran estas direcciones, deben investigarse más a fondo. Esta lista es solo para fines educativos y contiene ejemplos de direcciones IP privadas que se encuentran solo dentro de las redes internas.

Muestra esta lista y examina lo que contiene al ejecutarse la celda. Asegúrate de reemplazar el fragmento `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda.

[10]:
```
# Asigna a `flagged_addresses` una lista de direcciones IP que se denunciaron␣
↪previamente por actividad inusual
```

```
flagged_addresses = ["192.168.190.178", "192.168.96.200", "192.168.174.117",␣
 ↪"192.168.168.144"]

# Muestra el contenido de `flagged_addresses`

print(flagged_addresses)
```

```
['192.168.190.178', '192.168.96.200', '192.168.174.117', '192.168.168.144']
```

Pista 1

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

Tarea 11 Por último, escribirás una sentencia iterativa que recorre la lista `valid_ip_addresses` y comprueba si cada dirección IP está denunciada. En el siguiente código, `address` será la variable de bucle. Además, incluye una sentencia condicional que comprueba si `address` pertenece a la lista `flagged_addresses`. Si es así, debería mostrar `"Se marcó la dirección IP ______ para su posterior análisis."` Si no es así, debería mostrar `"La dirección IP ______ no requiere un análisis adicional."`Asegúrate de reemplazar cada `### YOUR CODE HERE ###` con tu propio código antes de ejecutar la siguiente celda.

[11]:
```
# Asigna a `log_file` una cadena que contiene el nombre de usuario, la fecha,␣
 ↪la hora de inicio de sesión y la dirección IP para una serie de intentos de␣
 ↪inicio de sesión

log_file = "eraab 2022-05-10 6:03:41 192.168.152.148 \niuduike 2022-05-09 6:46:
 ↪40 192.168.22.115 \nsmartell 2022-05-09 19:30:32 192.168.190.178 \narutley␣
 ↪2022-05-12 17:00:59 1923.1689.3.24 \nrjensen 2022-05-11 0:59:26 192.168.213.
 ↪128 \naestrada 2022-05-09 19:28:12 1924.1680.27.57 \nasundara 2022-05-11 18:
 ↪38:07 192.168.96.200 \ndkot 2022-05-12 10:52:00 1921.168.1283.75 \nabernard␣
 ↪2022-05-12 23:38:46 19245.168.2345.49 \ncjackson 2022-05-12 19:36:42 192.168.
 ↪247.153 \njclark 2022-05-10 10:48:02 192.168.174.117 \nalevitsk 2022-05-08␣
 ↪12:09:10 192.16874.1390.176 \njrafael 2022-05-10 22:40:01 192.168.148.115␣
 ↪\nyappiah 2022-05-12 10:37:22 192.168.103.10654 \ndaquino 2022-05-08 7:02:35␣
 ↪192.168.168.144"

# Asigna `pattern` a una expresión regular que coincida con todas las␣
 ↪direcciones IP válidas y solo aquellas

pattern = "\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}"

# Utiliza `re.findall()` en `pattern` y `log_file` y asigna␣
 ↪`valid_ip_addresses` al resultado

valid_ip_addresses = re.findall(pattern, log_file)

# Asigna a `flagged_addresses` una lista de direcciones IP que se denunciaron␣
 ↪previamente por actividad inusual
```

```
flagged_addresses = ["192.168.190.178", "192.168.96.200", "192.168.174.117",␣
↪"192.168.168.144"]

# La sentencia iterativa comienza aquí
# Bucle a través de `valid_ip_addresses` con `address` como variable de bucle

for address in valid_ip_addresses:

    # Condicional comienza aquí
    # Si `address` pertenece a `flagged_addresses`, muestra "Se denunció la␣
↪dirección IP ______ para un análisis adicional".

    if address in flagged_addresses:
        print("La dirección IP", address, "se denunció para un análisis␣
↪adicional.")

    # De lo contrario, muestra "La dirección IP ______ no requiere un análisis␣
↪adicional".

    else:
        print("La dirección IP", address, "no requiere un análisis adicional.")
```

```
La dirección IP 192.168.152.148 no requiere un análisis adicional.
La dirección IP 192.168.22.115 no requiere un análisis adicional.
Se denunció la dirección IP 192.168.190.178 para un análisis adicional.
La dirección IP 192.168.213.128 no requiere un análisis adicional.
Se denunció la dirección IP 192.168.96.200 para un análisis adicional.
La dirección IP 192.168.247.153 no requiere un análisis adicional.
Se denunció la dirección IP 192.168.174.117 para un análisis adicional.
La dirección IP 192.168.148.115 no requiere un análisis adicional.
La dirección IP 192.168.103.106 no requiere un análisis adicional.
Se denunció la dirección IP 192.168.168.144 para un análisis adicional.
```

Pista 1

Completa la condición de bucle `for` para que el bucle se repita a través de la lista `valid_ip_addresses`.

Pista 2

Completa la condición `if` para que la sentencia `if` compruebe si el valor de la variable de bucle `address` está en la lista `flagged_addresses`.

Pista 3

Dentro de la sentencia `else`, utiliza la función `print()` para mostrar el mensaje especificado.

1.9 Conclusión

¿A qué conclusiones clave llegaste con este laboratorio?

— Las expresiones regulares en Python te permiten crear patrones que luego puedes utilizar para encontrar cadenas importantes. — Se pueden construir patrones de expresión regular para que coincidan con caracteres específicos y combinaciones de caracteres. — Ejemplos de símbolos de expresión regular practicados en este laboratorio: - `\w` representa cualquier carácter alfanumérico. — `+` representa una o más apariciones del carácter anterior en la expresión regular. — `\d` representa cualquier dígito. — `\.` representa un punto. — `{x,y}` representa el número de apariciones en cualquier lugar entre x e y del carácter anterior en la expresión regular. La x y la y se pueden reemplazar con dos números enteros positivos para indicar un rango exacto para el número de ocurrencias. — La biblioteca `re` en Python contiene funciones que son útiles cuando se trabaja con expresiones regulares. — Un ejemplo es la función `re.findall()`, que toma un patrón de expresión regular, así como una cadena, verifica todas las instancias de la cadena que coinciden con el patrón y emite una lista de las coincidencias.