Activity_Import and parse a text file

April 11, 2024

1 Activity: Import and parse a text file

1.1 Introduction

Security logs are often stored in text files. To analyze the security logs in these files, security analysts have to import and parse these files. Python has some functions that come in handy for these tasks, allowing analysts to efficiently access information from text files.

1.2 Scenario

In this lab, I'm working as a security analyst. I am responsible for preparing a security log file for analysis and creating a text file with IP addresses that are allowed to access restricted information.

1.3 Task

In this task, I'll import a security log text file and store it as a string to prepare it for analysis.

In Python, a `with` statement is often used in file handling to open a file and then automatically close the file after reading it. I am given a variable named `import_file` that contains the name of the log file that you want to import. I'll start by writing the first line of the `with` statement. I'll use the `open()` function, setting the second parameter to `"r"`.

[ ]:
```
# Assign `import_file` to the name of the text file that contains the security␣
↪log file

import_file = "login.txt"

# First line of the `with` statement
# Use `open()` to import security log file and store it as a string

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

the second parameter takes in a string that indicates how the file should be handled. The letter `"r"` is used as the second argument to read a file

1.4 Task

I'll use the `.read()` method to read the imported file, and store the result in a variable named `text`.

[3]:
```
# Assign `import_file` to the name of the text file that contains the security log file

import_file = "login.txt"

# The `with` statement
# Use `open()` to import security log file and store it as a string

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

  # Use `.read()` to read the imported file and store the result in a variable named `text`

  text = file.read()

# Display the contents of `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
```

1.5 Task

The output in the previous step is one big string. In this task, I'll explore how to split the string that contains the entire imported log file into a list of strings, one string per line. I'll use the `.split()` method to perform this split and then display the result.

[5]:
```
# Assign `import_file` to the name of the text file that contains the security␣
 ↪log file

import_file = "login.txt"

# The `with` statement
# Use `open()` to import security log file and store it as a string

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

  # Use `.read()` to read the imported file and store the result in a variable␣
 ↪named `text`

  text = file.read()

# Display the contents of `text` split into separate lines

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']
```

The `.split()` method in Python converts a string into a list. It can take in a separator character that specifies which character to split on. If a character is not specified, it will split on whitespace

by default.

1.6 Task

There is a missing entry in the log file. I'll need to account for that by appending it to the log file. I'm given the missing entry stored in a variable named `missing_entry`.

I'll use the `.write()` method and the parameter `"a"` in the `open()` function.

[7]:
```
# Assign `import_file` to the name of the text file that contains the security␣
 ↪log file

import_file = "login.txt"

# Assign `missing entry` to a log that was not recorded in the log file

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

# Use `open()` to import security log file and store it as a string
# Pass in "a" as the second parameter to indicate that the file is being opened␣
 ↪for appending purposes

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

    # Use `.write()` to append `missing_entry` to the log file

    file.write(missing_entry)

# Use `open()` with the parameter "r" to open the security log file for reading␣
 ↪purposes

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

    # Use `.read()` to read in the contents of the log file and store in a␣
 ↪variable named `text`

    text = file.read()

# Display the contents of `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,192.168.243.140,4:56:27,2022-05-09
```

1.7 Task

The next task is creating a text file. This text file should include a list of IP addresses that are allowed to access restricted information. Documenting this in a text file will help communicate findings to the security team.

I'll start by creating a variable named `import_file` that stores the name of the file, which should be `"allow_list.txt"`. I'm also given a variable named `ip_addresses` that stores a string containing the IP addresses that are allowed.

[8]:
```
# Assign `import_file` to the name of the text file that you want to create

import_file = "allow_list.txt"

# Assign `ip_addresses` to a list of IP addresses that are allowed to access␣
 ↪the restricted information

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"

# Display `import_file`

print(import_file)

# Display `ip_addresses`

print(ip_addresses)
```

```
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
```

1.8 Task

The next goal is to create a `with` statement in order to write the IP addresses to the text file created in the previous step.

I'll first open the file using the `"w"` parameter. Then, write the IP addresses to the file.

[11]:
```
# Assign `import_file` to the name of the text file that you want to create

import_file = "allow_list.txt"

# Assign `ip_addresses` to a list of IP addresses that are allowed to access␣
 ↪the restricted information

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"

# Create a `with` statement to write to the text file

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

  # Write `ip_addresses` to the text file

  file.write(ip_addresses)
```

1.9 Task

In this final step, I'll complete the code I've been writing. I'll add code to read the file containing IP addresses. I'll complete a `with` statement that reads the text file and stores it in a new variable called `text`.

[12]:
```
# Assign `import_file` to the name of the text file that you want to create

import_file = "allow_list.txt"

# Assign `ip_addresses` to a list of IP addresses that are allowed to access␣
 ↪the restricted information

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"
```

```
# Create a `with` statement to write to the text file

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

    # Write `ip_addresses` to the text file

    file.write(ip_addresses)

# Create a `with` statement to read in the text file

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

    # Read the file and store the result in a variable named `text`

    text = file.read()

# Display the contents of `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
```