Malware-Analysis-Scripts/deobfuscate_ostap.py at master · cryptogramfan/Malware-Analysis-Scripts · GitHub

github.com/cryptogramfan/Malware-Analysis-Scripts/blob/master/deobfuscate_ostap.py

cryptogramfan

cryptogramfan/**Malware-Analysis-Scripts**

Handy scripts to speed up malware analysis

1 Contributor 0 Issues 32 Stars 4 Forks

```
#!/usr/bin/env python
#
# A script that deobfuscates Ostap JSE (JScript Encoded) downloaders. The script is based
# on Ostap samples analysed in August 2019, such as those delivering TrickBot. It will try
# to identify the indexes containing Unicode character codes and then deobfuscate the sample
# using subtraction and addition.
#
# To use the script, supply a file as an argument or pipe it to stdin:
#
# $ python deobfuscate_ostap.py ostap.jse
# $ cat ostap.jse | deobfuscate_ostap.py
#
# Author.....: Alex Holland (@cryptogramfan)
# Date.......: 2019-08-29
# Version....: 0.0.5
# License....: CC BY 4.0
# Reference_1: https://www.bromium.com/deobfuscating-ostap-trickbots-javascript-downloader/

import os
import sys
import re

index_0 = ""
index_1 = ""
indexes_raw = []
indexes = []
values_0 = []
values_1 = []

# Subtract index 0 values from index 1
def subtract_values_1():
characters_sub = []
servers = []
urls = []

try:
print("[+] Trying deobfuscation by subtracting index %s elements from index %s elements..." % (indexes[0], indexes[1]))
```

```
charcodes_sub = [i - j for i, j in zip(values_0, values_1)]

except:
print("[!] Error subtracting index %s elements from index %s elements." % (indexes[0],
indexes[1]))
subtract_values_2() # Try another subtraction instead

try:
for charcode_sub in charcodes_sub:
character_sub = chr(charcode_sub)
characters_sub.append(character_sub)

characters_sub = ''.join(characters_sub)

except:
print("[!] Error converting character codes to characters.")
subtract_values_2()

match = re.search("Script", characters_sub, re.IGNORECASE)

if match:
print("[+] Deobfuscation using subtraction 1 was successful:\n")
print(characters_sub)

match_url = re.search("http(s):\/\/.+(Drives|POST)", characters_sub, re.IGNORECASE)

if match_url:
servers.append(match_url.group())

for server in servers:
server = re.sub("Drives.*$", "", server, re.IGNORECASE)
server = re.sub("POST$", "", server, re.IGNORECASE)
```

```
urls.append(server)

if urls:
print("\n[+] Found URL(s):\n")
print(", ".join(urls))

exit(0)

else:
print("[!] Deobfuscation using subtraction 1 was unsuccessful.")
subtract_values_2()

return;

# Subtract index 1 values from index 0 values
def subtract_values_2():
characters_sub = []
servers = []
urls = []

try:
print("[+] Trying deobfuscation by subtracting index %s elements from index %s elements..." % (indexes[1], indexes[0]))
charcodes_sub = [i - j for i, j in zip(values_1, values_0)]

except:
print("[!] Error subtracting index %s elements from index %s elements." % (indexes[1], indexes[0]))
add_values() # Try addition instead

try:
for charcode_sub in charcodes_sub:
```

```
character_sub = chr(charcode_sub)
characters_sub.append(character_sub)

characters_sub = ".join(characters_sub)

except:
print("[!] Error converting character codes to characters.")
add_values()

match = re.search("Script", characters_sub, re.IGNORECASE)

if match:
print("[+] Deobfuscation using subtraction 2 was successful:\n")
print(characters_sub)

match_url = re.search("http(s):\/\/.+(Drives|POST)", characters_sub, re.IGNORECASE)

if match_url:
servers.append(match_url.group())

for server in servers:
server = re.sub("Drives.*$", "", server, re.IGNORECASE)
server = re.sub("POST$", "", server, re.IGNORECASE)
urls.append(server)

if urls:
print("\n[+] Found URL(s):\n")
print(", ".join(urls))

exit(0)

else:
```

```
print("[!] Deobfuscation using subtraction 2 was unsuccessful.")
add_values()

return;

# Add index 0 values to index 1 values
def add_values():
characters_add = []
servers = []
urls = []

try:
print("[+] Trying deobfuscation by adding index %s elements to index %s elements..." %
(indexes[1], indexes[0]))
charcodes_add = [i + j for i, j in zip(values_1, values_0)]

except:
print("[!] Error adding index %s elements to index %s elements. Exiting." % (indexes[1],
indexes[0]))
exit(0)

try:
for charcode_add in charcodes_add:
character_add = chr(charcode_add)
characters_add.append(character_add)

characters_add = ''.join(characters_add)

except:
print("[!] Error converting character codes to characters. Exiting.")
exit(0)
```

```
match = re.search("Script", characters_add, re.IGNORECASE)

if match:
print("[+] Deobfuscation using addition was successful:\n")
print(characters_add)

match_url = re.search("http(s):\/\/.+(Drives|POST)", characters_add, re.IGNORECASE)

if match_url:
servers.append(match_url.group())

for server in servers:
server = re.sub("Drives.*$", "", server, re.IGNORECASE)
server = re.sub("POST$", "", server, re.IGNORECASE)
urls.append(server)

if urls:
print("\n[+] Found URL(s):\n")
print(", ".join(urls))

exit(0)

else:
print("[!] Deobfuscation using addition was unsuccessful. Exiting.")
exit(0)

return;

if len(sys.argv) > 1:
file = open(sys.argv[1], 'r')
else:
```

```
file = sys.stdin

while 1:
input = file.read()

# Find array indexes
try:
print("\n[+] Analysing %s" % os.path.basename(file.name))
input = input.decode('utf-8')

except UnicodeError:
print("[!] File not UTF-8. Treating as UTF-16.")
input = input.decode('utf-16')

try:
indexes_raw = re.findall("\[\d+\]=\d+;", input)

except:
print("[!] Error finding array indexes. Exiting.")
exit(0)

if not indexes_raw:
print("[!] Array indexes not found. Exiting.")
exit(0)

# Put the index string into a list
try:
for index in indexes_raw:
index = re.sub("\[", "", index)
index = re.sub("\]=\d+;", "", index)
```

```
indexes.append(index)

# Remove duplicates
indexes = list(set(indexes))
print("[+] Found array indexes %s and %s." % (indexes[0], indexes[1]))

except:
print("[!] Error processing array indexes. Exiting.")
exit(0)

try:
element_regex_0 = r"\[" + indexes[0] + r"\]=\d+;"
element_regex_1 = r"\[" + indexes[1] + r"\]=\d+;"

except:
print("[!] Error creating regular expressions. Exiting.")
exit(0)

# Find the values of index 0 elements
try:
print("[+] Searching for index %s elements..." % indexes[0])
array_0 = re.findall(element_regex_0, input)

for element in array_0:
element = re.sub("\[\d+\]=", "", element)
element = re.sub(";", "", element)
values_0.append(element)

except:
print("[!] Error finding index %s elements. Exiting." % indexes[0])
```

```
exit(0)

if not values_0:
print("[!] No index %s elements found. Exiting." % indexes[0])
exit(0)

# Convert index 0 elements to integer values
try:
values_0 = map(int, values_0)

except:
print("[!] Error converting index %s elements to integers. Exiting." % indexes[0])
exit(0)

# Find the values of index 1 elements
try:
print("[+] Searching for index %s elements..." % indexes[1])
array_1 = re.findall(element_regex_1, input)

for element in array_1:
element = re.sub("\[\d+\]=", "", element)
element = re.sub(";", "", element)
values_1.append(element)

except:
print("[!] Error finding index %s elements. Exiting." % indexes[1])
exit(0)

if not values_1:
print("[!] No index %s elements found. Exiting." % indexes[1])
```

```
exit(0)

# Convert index 1 elements to integer values
try:
values_1 = map(int, values_1)

except:
print("[!] Error converting index %s elements to integers. Exiting." % indexes[1])
exit(0)

subtract_values_1()
subtract_values_2()
add_values()

exit(0)
```