hw08_Panchagnula_Raghava

Homework Number: HW08

Name: Raghava Vivekananda Panchagnula

ECN Login: rpanchag

Due Date: 3/21/2024

Command to run py file

```
(.venv) vivek@vivek-desktop:~/Files/coursework/ECE-40400/Homework/HW08$ make test
sudo /home/vivek/Files/.venv/bin/python3.10 TcpAttack.py
```

Code uses functions given in lecture notes, abiding instructions given in HW document.

```
(.venv) vivek@vivek-desktop:~/Files/coursework/ECE-40400/Homework/HW08$ make tcp
sudo tcpdump -vvv -nn -s 1500 -S -X -c 5 'dst moonshine.ecn.purdue.edu'
tcpdump: listening on wlo1, link-type EN10MB (Ethernet), snapshot length 1500 bytes
17:19:12.274386 IP (tos 0x0, ttl 64, id 8958, offset 0, flags [DF], proto TCP (6), length 60)
    10.0.0.91.35130 > 128.46.144.123.1700: Flags [S], cksum 0x1b33 (incorrect -> 0xecb0), seq 1517269427, win 64240, options [mss 1460,sackOK,TS val
3568948382 ecr 0,nop,wscale 7], length 0
        0x0000:  4500 003c 22fe 4000 4006 fcb9 0a00 005b  E..<".@.@......[
        0x0010:  802e 907b 893a 06a4 5a6f b1b3 0000 0000  ...{.:..Zo......
        0x0020:  a002 faf0 1b33 0000 0204 05b4 0402 080a  .....3..........
        0x0030:  d4b9 d49e 0000 0000 0103 0307            ............
17:19:12.307023 IP (tos 0x0, ttl 64, id 17305, offset 0, flags [DF], proto TCP (6), length 60)
    10.0.0.91.36670 > 128.46.144.123.1701: Flags [S], cksum 0x1b33 (incorrect -> 0x041d), seq 3897951802, win 64240, options [mss 1460,sackOK,TS val
3568948415 ecr 0,nop,wscale 7], length 0
        0x0000:  4500 003c 4399 4000 4006 dc1e 0a00 005b  E..<C.@.@......[
        0x0010:  802e 907b 8f3e 06a5 e856 063a 0000 0000  ...{.>...V.:....
        0x0020:  a002 faf0 1b33 0000 0204 05b4 0402 080a  .....3..........
        0x0030:  d4b9 d4bf 0000 0000 0103 0307            ............
17:19:12.332808 IP (tos 0x0, ttl 64, id 51630, offset 0, flags [DF], proto TCP (6), length 60)
    10.0.0.91.46516 > 128.46.144.123.1702: Flags [S], cksum 0x1b33 (incorrect -> 0x530c), seq 2985543452, win 64240, options [mss 1460,sackOK,TS val
3568948441 ecr 0,nop,wscale 7], length 0
```

TCPdump output when scanning ports

```
(.venv) vivek@vivek-desktop:~/Files/coursework/ECE-40400/Homework/HW08$ make tcp
sudo tcpdump -vvv -nn -s 1500 -S -X -c 20 'dst moonshine.ecn.purdue.edu'
tcpdump: listening on wlo1, link-type EN10MB (Ethernet), snapshot length 1500 bytes
17:24:55.953155 IP (tos 0x0, ttl 64, id 1, offset 0, flags [none], proto TCP (6), length 40)
    10.10.10.10.63416 > 128.46.144.123.1716: Flags [S], cksum 0x6cb8 (correct), seq 0, win 8192, length 0
        0x0000:  4500 0028 0001 0000 4006 5612 0a0a 0a0a  E..(....@.V.....
        0x0010:  802e 907b f7b8 06b4 0000 0000 0000 0000  ...{............
        0x0020:  5002 2000 6cb8 0000                      P...l...
17:24:55.989175 IP (tos 0x0, ttl 64, id 1, offset 0, flags [none], proto TCP (6), length 40)
    10.10.10.10.64775 > 128.46.144.123.1716: Flags [S], cksum 0x6769 (correct), seq 0, win 8192, length 0
        0x0000:  4500 0028 0001 0000 4006 5612 0a0a 0a0a  E..(....@.V.....
        0x0010:  802e 907b fd07 06b4 0000 0000 0000 0000  ...{............
        0x0020:  5002 2000 6769 0000                      P...gi..
17:24:56.033103 IP (tos 0x0, ttl 64, id 1, offset 0, flags [none], proto TCP (6), length 40)
    10.10.10.10.60652 > 128.46.144.123.1716: Flags [S], cksum 0x7784 (correct), seq 0, win 8192, length 0
        0x0000:  4500 0028 0001 0000 4006 5612 0a0a 0a0a  E..(....@.V.....
        0x0010:  802e 907b ecec 06b4 0000 0000 0000 0000  ...{............
        0x0020:  5002 2000 7784 0000                      P...w...
```

TCPDump output when SYN flood attacking on port 1716

```
0 packets dropped by kernel
(.venv) vivek@vivek-desktop:~/Files/coursework/ECE-40400/Homework/HW08$ make subm
zip -r hw08_Panchagnula_Raghava.zip hw08_Panchagnula_Raghava.pdf TcpAttack.py
        zip warning: name not matched: hw08_Panchagnula_Raghava.pdf
  adding: TcpAttack.py (deflated 74%)
(.venv) vivek@vivek-desktop:~/Files/coursework/ECE-40400/Homework/HW08$
```

Commands to create the output (ignore zip warning as PDF wasn't created for this example.)

Make file as below:

```
##Static Variables
HW_NUM = 08

TEST_PYFILE = TcpAttack_test.py
PYFILE = TcpAttack.py
PYTHON = /home/vivek/Files/.venv/bin/python3.10

##Targets
test: attack

attack:
    . $(VENV)
    sudo $(PYTHON) $(TEST_PYFILE)

tcp:
    sudo tcpdump -vvv -nn -s 1500 -S -X -c 20 'dst moonshine.ecn.purdue.edu'

submit:
    zip -r hw$(HW_NUM)_Panchagnula_Raghava.zip hw$(HW_NUM)_Panchagnula_Raghava.pdf $(PYFILE)
```

- Preamble:
  - Import scapy and socket, define TCPAttack class
  - The lecture code works almost entirely on its own with very little modification, I stated the modifications I made below, but not much else was changed.
- Summary of TcpAttack class:
  - Init
    - Initializes spoof ip and target ip
  - scanTarget
    - Largely adapted from avi kak's lecture 16 code
    - It takes parameters for the start and end of the port range to be scanned. The code iterates through each port in the specified range, attempting to establish a TCP connection. If successful, it adds the open port to a list. Additionally, it attempts to identify services associated with the open ports by referencing the /etc/services file. The results are written to a file named "openports.txt"

        - It uses socket.connect in a try, except loop. This could be done with connect_ex but this works too, and since it was done like this in the lecture code, I left it as is.
        - Added a sock.close() to the try and except cases so that If a socket is found, it is closed correctly to let the next port be found.
        - The lecture code is taken, it uses regex to look through service ports to figure out the purpose of each port
    - attackTarget
        - It takes parameters for the target port and the number of SYN packets to send. The code attempts to establish a TCP connection to the target port, and if successful, it sends the specified number of SYN packets with spoofed source IP addresses using the Scapy library. Finally, it returns 1 if the port was open and 0 if it was closed.
        - It uses connect_ex instead of connect like in lecture notes because connect_ex doesn't raise an exception when it fails, and it instead just returns 0 if succeeded.
        - It checks if the port is open using socket, same method used in scanTarget and in lecture notes
        - If the socket is open, generate an ip header and tcp header, and send it. Return exception if not.
- main
    - scans from port 1000 to 2000, and checks for open ports, and writes open ports to openports.txt
    - attacks ports listed in openports.txt with 100 SYN packets
    - Tested LAN ip at home with port 2000 open with a terraria server, mixed results, but it for sure slowed down the connection by quite a bit…
    - Possible that the Terraria server implements some sort of fail2ban or some other protection by itself…