Department of Computer Science & Engineering
Computer Networks
UE20CS253

Name	SRN
Shubham S	PES1UG20CS420
Siddarth M P	PES1UG20CS423
Pavankumar Hegde	PES1UG20CS823

" TCP PORT SCANNING USING NETCAT AND OTHER FUNCTIONAL APPLICATIONS" :

ABSTRACT :

This project details many of the techniques used to determine what ports (or similar protocol abstraction) of a host are listening for connections. These ports represent potential communication channels. Netcat is a command-line network tool used to open ports, associate a shell to a port, establish TCP/UDP connections, and more. This Project shows how to use Netcat to scan ports on remote targets. Examples include explaination on individual port scan, scanning several ports, scanning port ranges, and Netcat can also be used directly with other programs and scripts to send files from a client to a server and back , It can be used to **relay information from one port on a specific machine to another port on a different machine**

PROJECT EXPLANATION :

In the below project, we have implemented TCP Port scanner, using Nmap.

Initially all of 1000 ports (0 to 999) will be closed, we used Netcat to

manually open a port, note that port will be opened only when an application is being run i.e, it needs to respond upon client's requests which occurs only when the port will be open and then upon the completion of the transaction, the port will be closed, in our case the Transaction is just to manually open the port for scanning , we have implemented other functions under Netcat ,

1) **File transfer: here we specify receiver's IP and specific port number by which we can send the files across the 2 systems, hence supporting multi-clients.**
2) **Message Passing: Here we can send the messages/text information between 2 systems dynamically and retrieve the same on the receiver's side.**

PORT SCANNING USING NMAP :

CODE:

SERVER SIDE :

```
from socket import *
from datetime import *
from datetime import datetime
import pytz
import nmap

begin = 1
end = 900
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind(("10.0.2.8",serverPort))
print("The server is ready to receive")


m=input("1. Default Connected Client:\n2. Enter Host :")
```

```
while 1:
    if m == '1':
     while 1:
            print("Select Option :")


            serverSocket.listen(1)

            connectionSocket, addr = serverSocket.accept()

            sentence = connectionSocket.recv(1024)
            print ("")

            client1='10.0.2.13'
            client2=''
            client3=''


            if client1 in connectionSocket.getpeername() :
                    print ("Client 1 :",client1)
                    target=client1
            #elif client2 in connectionSocket.getpeername():
            #       print ("Client 2 :")
            #       target=client2
            #elif client3 in connectionSocket.getpeername():
            #       print ("Client 3 :")
            #       target=client3

            scanner = nmap.PortScanner()

            for i in range(20,440):
                    res = scanner.scan(target,str(i))

                    res = res['scan'][target]['tcp'][i]['state']

                    print("port ",i," is "+res)
            print ('Server Responding To Client IP Address
    :\n',connectionSocket.getpeername())
```

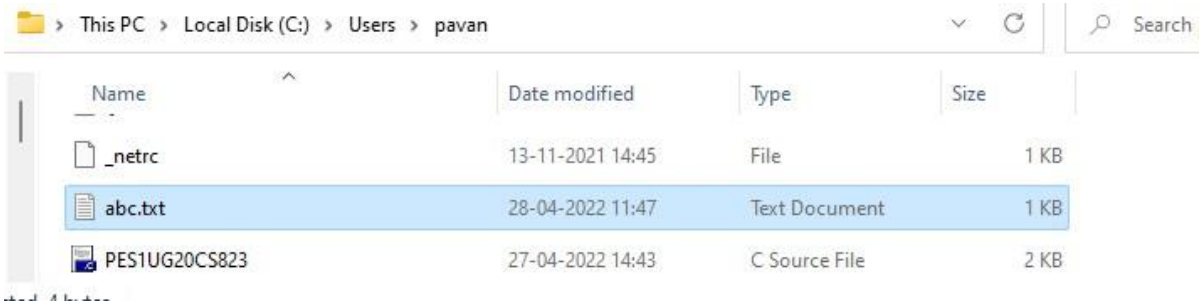
```
        #('Time Sent From Server')
        #now = datetime.now()
        #current_time = now.strftime("%H:%M:%S")
        #current_time1 = now.strftime("%D")
        #current_time2 = now.strftime("%c")
        #date="Time :"+current_time+" on Date :"+current_time1+ "
    Extra Data:  "+ current_time2


        #connectionSocket.send(date.encode())

        #print(current_time)



     connectionSocket.close()

    elif m=='2':
     print("i")
     j=input("Enter the IP: ")
     scanner = nmap.PortScanner()

     low=int(input("Enter the lower port range: "))
     high=int(input("Enter the higer port ranger: "))
     for i in range(low,high):
        res = scanner.scan(j,str(i))

        res = res['scan'][j]['tcp'][i]['state']

            print("port ",i," is "+res)
```

CLIENT CODE :

```
from socket import *
serverName = "10.0.2.8"
serverPort = 12000while True:
    clientSocket = socket(AF_INET, SOCK_STREAM)
    clientSocket.connect((serverName,serverPort))

    sentence = input("Request for time ? (Y/y or any key to exit) :")
    if sentence == "Y" or sentence == "y" :
```

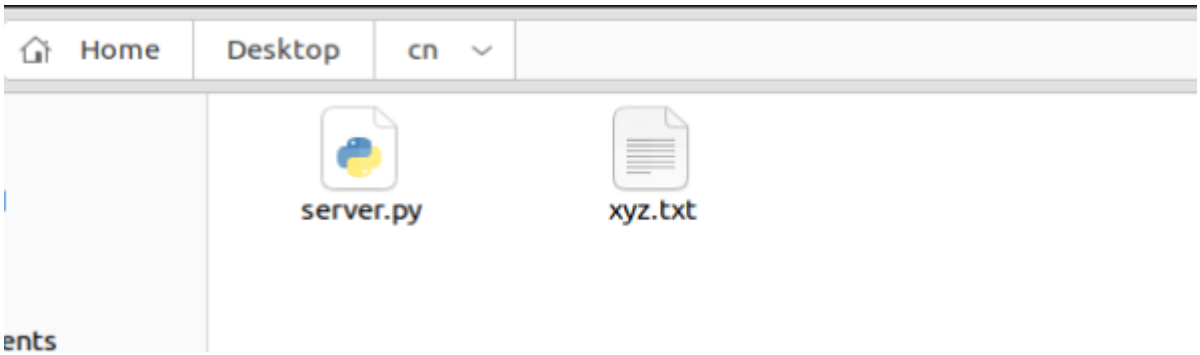
```
        print (" ")
        clientSocket.sendto(sentence.encode(),(serverName, serverPort))
        modifiedSentence = clientSocket.recv(1024)
        print("From Server:", modifiedSentence)
        clientSocket.close()

    else:
        break;
```

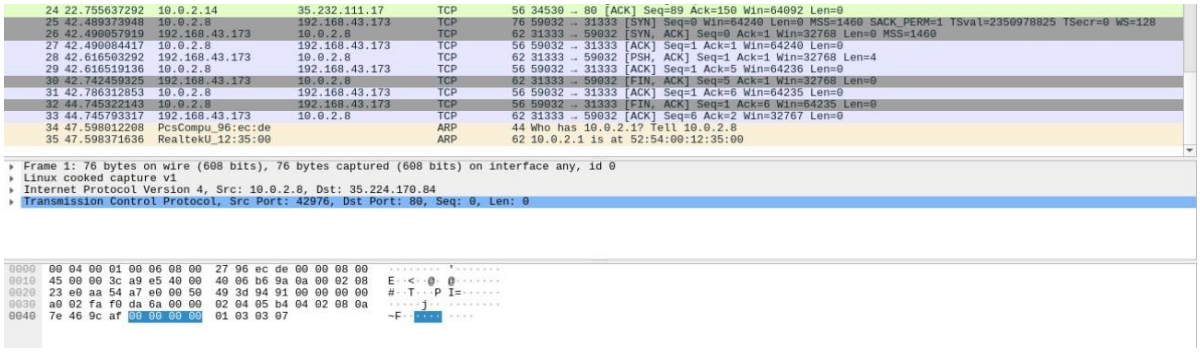
File Transfer Using Netcat ,



```
C:\Users\pavan>ncat -v -n -w 30 -p 31333 -l < abc.txt.txt
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::31333
Ncat: Listening on 0.0.0.0:31333
Ncat: Connection from 192.168.43.173.
Ncat: Connection from 192.168.43.173:51073.
```

```
server@PES1UG20CS823:~/Desktop/cn$ sudo nc -v -w 2 192.168.43.173 31333 > xyz.txt
Connection to 192.168.43.173 31333 port [tcp/*] succeeded!
```





24	22.755637292	10.0.2.14	35.232.111.17	TCP	56 34530 → 80 [ACK] Seq=89 Ack=150 Win=64092 Len=0
25	42.489373948	10.0.2.8	192.168.43.173	TCP	76 59032 → 31333 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2350978825 TSecr=0 WS=128
26	42.490057919	192.168.43.173	10.0.2.8	TCP	62 31333 → 59032 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=1460
27	42.490084417	10.0.2.8	192.168.43.173	TCP	56 59032 → 31333 [ACK] Seq=1 Ack=1 Win=64240 Len=0
28	42.616503292	192.168.43.173	10.0.2.8	TCP	62 31333 → 59032 [PSH, ACK] Seq=1 Ack=1 Win=32768 Len=4
29	42.616519136	10.0.2.8	192.168.43.173	TCP	56 59032 → 31333 [ACK] Seq=1 Ack=5 Win=64236 Len=0
30	42.742459325	192.168.43.173	10.0.2.8	TCP	62 31333 → 59032 [FIN, ACK] Seq=5 Ack=1 Win=32768 Len=0
31	42.786312853	10.0.2.8	192.168.43.173	TCP	56 59032 → 31333 [ACK] Seq=1 Ack=6 Win=64235 Len=0
32	44.745322143	10.0.2.8	192.168.43.173	TCP	56 59032 → 31333 [FIN, ACK] Seq=1 Ack=6 Win=64235 Len=0
33	44.745793317	192.168.43.173	10.0.2.8	TCP	62 31333 → 59032 [ACK] Seq=6 Ack=2 Win=32767 Len=0
34	47.598012208	PcsCompu_96:ec:de		ARP	44 Who has 10.0.2.1? Tell 10.0.2.8
35	47.598371636	RealtekU_12:35:00		ARP	62 10.0.2.1 is at 52:54:00:12:35:00

```
▸ Frame 1: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface any, id 0
▸ Linux cooked capture v1
▸ Internet Protocol Version 4, Src: 10.0.2.8, Dst: 35.224.170.84
▸ Transmission Control Protocol, Src Port: 42976, Dst Port: 80, Seq: 0, Len: 0

0000  00 04 00 01 00 06 08 00  27 96 ec de 00 00 08 00   ........ '.......
0010  45 00 00 3c a9 e5 40 00  40 06 b6 9a 0a 00 02 08   E··<··@· @·......
0020  23 e0 aa 54 a7 e0 00 50  49 3d 94 91 00 00 00 00   #··T···P I=......
0030  a0 02 fa f0 da 6a 00 00  02 04 05 b4 04 02 08 0a   ·····j·· ........
0040  7e 46 9c af 00 00 00 00  01 03 03 07               ~F······ ····
```

Message Passing Using Netcat

```
server@PES1UG20CS823:~/Desktop/cn$ sudo nc -l 3260
hi hello
```

```
pavankumar@PES1UG20CS823:~/Desktop/cn$ sudo nc 10.0.2.8 3260
hello
hi
```

No.	Time	Source	Destination	Protocol	Length	Info
58	225.581866204	10.0.2.13	10.0.2.8	TCP	76	44126 → 3260 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2137293132 TSecr=0 WS=128
59	225.581886817	10.0.2.8	10.0.2.13	TCP	76	3260 → 44126 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=1686893688 TSecr=…
60	225.582431180	10.0.2.13	10.0.2.8	TCP	68	44126 → 3260 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2137293132 TSecr=1686893688
61	225.582513590	10.0.2.8	10.0.2.13	TCP	78	3260 → 44126 [PSH, ACK] Seq=1 Ack=1 Win=65280 Len=10 TSval=1686893689 TSecr=2137293132
62	225.583860296	10.0.2.13	10.0.2.8	TCP	68	44126 → 3260 [ACK] Seq=1 Ack=11 Win=64256 Len=0 TSval=2137293134 TSecr=1686893689
63	233.424599256	10.0.2.8	10.0.2.3	DHCP	335	DHCP Request - Transaction ID 0xa4e24f0f

```
▸ Frame 58: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface any, id 0
▸ Linux cooked capture v1
▸ Internet Protocol Version 4, Src: 10.0.2.13, Dst: 10.0.2.8
▸ Transmission Control Protocol, Src Port: 44126, Dst Port: 3260, Seq: 0, Len: 0

0000  00 00 00 01 00 06 08 00  27 b5 9b b8 00 00 08 00   ........ '.......
0010  45 00 00 3c 71 74 40 00  40 06 b1 33 0a 00 02 0d   E··<qt@· @··3····
0020  0a 00 02 08 ac 5e 0c bc  6d ba f2 cb 00 00 00 00   ·····^·· m·......
0030  a0 02 fa f0 1a a9 00 00  02 04 05 b4 04 02 08 0a   ........ ........
0040  7f 64 81 4c 00 00 00 00  01 03 03 07               ·d·L···· ····
```

Manually opening Ports Using Netcat

```
pavankumar@PES1UG20CS823:~$ sudo nc -lk 68

GET / HTTP/1.0

OPTIONS / HTTP/1.0

OPTIONS / RTSP/1.0

◆(r◆◆◆◆|versionbind
                    HELP
SO?G◆◆◆,◆◆`~◆◆◆{◆%◆w◆◆◆◆<=◆o◆n(
fedcba`*%◆Cookie: mstshash=nmap
leU◆◆random1random2random3random4
                                /
,*qj◆n0◆k◆◆
◆◆^0\◆P◆◆M◆0◆◆0
          rbtgtM◆19700101000000Z◆◆A0◆◆SMB@@◆PC NETWORK PROGRAM 1.0MICROSOFT NETWORKS 1.03MICROSOFT NETWORKS 3.0LAN
MAN1.0LM1.2X002SambaNT LANMAN 1.0NT LM 0.12l
     GET /nice%20ports%2C/Tri%6Eity.txt%2ebak HTTP/1.0

default
0◆-c◆$

d◆
  objectClass0◆0
                `◆OPTIONS sip:nm SIP/2.0
Via: SIP/2.0/TCP nm;branch=foo
From: <sip:nm@nm>;tag=root
To: <sip:nm2@nm2>
Call-ID: 50000
CSeq: 42 OPTIONS
Max-Forwards: 70
Content-Length: 0
Contact: <sip:nm@nm>
Accept: application/sdp

TNMPTNME
        ◆DmdT◆◆:/@=/@JRMIK◆◆◆
                             ◆◆MMS◆◆◆◆
```

No.	Time	Source	Destination	Protocol	Length	Info
189	94.490004433	10.0.2.8	10.0.2.13	TCP	68	60824 → 68 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1687191189 TSecr=2137590639
190	94.490247597	10.0.2.13	10.0.2.8	TCP	68	68 → 60822 [FIN, ACK] Seq=1 Ack=170 Win=65024 Len=0 TSval=2137590639 TSecr=1687191188
191	94.490269738	10.0.2.8	10.0.2.13	TCP	68	60822 → 68 [ACK] Seq=170 Ack=2 Win=64256 Len=0 TSval=1687191189 TSecr=2137590639
192	94.490366981	10.0.2.8	10.0.2.13	TCP	80	60824 → 68 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=12 TSval=1687191189 TSecr=2137590639
193	94.490815779	10.0.2.13	10.0.2.8	TCP	68	68 → 60824 [ACK] Seq=1 Ack=13 Win=65152 Len=0 TSval=2137590640 TSecr=1687191189
194	99.493444911	10.0.2.8	10.0.2.13	TCP	68	60824 → 68 [FIN, ACK] Seq=13 Ack=1 Win=64256 Len=0 TSval=1687196192 TSecr=2137590640
195	99.493672881	10.0.2.8	10.0.2.13	TCP	76	60826 → 68 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=1687196192 TSecr=0 WS=128
196	99.494798493	10.0.2.13	10.0.2.8	TCP	76	68 → 60826 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=2137595642 TSecr=16…
197	99.494798944	10.0.2.13	10.0.2.8	TCP	68	68 → 60824 [FIN, ACK] Seq=1 Ack=14 Win=65152 Len=0 TSval=2137595642 TSecr=1687196192
198	99.494877965	10.0.2.8	10.0.2.13	TCP	68	60826 → 68 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1687196194 TSecr=2137595642
199	99.495066499	10.0.2.8	10.0.2.13	TCP	68	60824 → 68 [ACK] Seq=14 Ack=2 Win=64256 Len=0 TSval=1687196194 TSecr=2137595642
200	99.495341419	10.0.2.8	10.0.2.13	TCP	121	60826 → 68 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=53 TSval=1687196194 TSecr=2137595642
201	99.496499601	10.0.2.13	10.0.2.8	TCP	68	68 → 60826 [ACK] Seq=1 Ack=54 Win=65152 Len=0 TSval=2137595644 TSecr=1687196194
202	104.497189490	10.0.2.8	10.0.2.13	TCP	68	60826 → 68 [FIN, ACK] Seq=54 Ack=1 Win=64256 Len=0 TSval=1687201196 TSecr=2137595644
203	104.497275155	10.0.2.8	10.0.2.13	TCP	76	60828 → 68 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=1687201196 TSecr=0 WS=128
204	104.497664592	10.0.2.13	10.0.2.8	TCP	76	68 → 60828 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=2137600644 TSecr=16…
205	104.497664734	10.0.2.13	10.0.2.8	TCP	68	68 → 60826 [FIN, ACK] Seq=1 Ack=55 Win=65152 Len=0 TSval=2137600644 TSecr=1687201196
206	104.497693527	10.0.2.8	10.0.2.13	TCP	68	60828 → 68 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1687201196 TSecr=2137600644

```
▸ Frame 193: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) on interface any, id 0
▸ Linux cooked capture v1
▸ Internet Protocol Version 4, Src: 10.0.2.13, Dst: 10.0.2.8
▸ Transmission Control Protocol, Src Port: 68, Dst Port: 60824, Seq: 1, Ack: 13, Len: 0

0000  00 00 00 01 00 06 08 00  27 b5 9b b8 08 00 08 00   ........ '.......
0010  45 00 00 34 cd 54 40 00  40 06 55 5b 0a 00 02 0d   E··4·T@· @·U[····
0020  0a 00 02 08 00 44 ed 98  22 ba aa 3e a7 4d 60 a7   ·····D·· "··>·M··
0030  80 10 01 fd 2b e2 00 00  01 01 08 0a 7f 69 0b 70   ····+··· ·····i·p
0040  64 90 7e 95                                        d·~·
```

Port Scanning Using Python-nmap module

```
server@PES1UG20CSB23:~/Desktop/cn$ python3 server.py
The server is ready to receive
1. Default Connected Client:
2. Enter Host :1
Select Option :

Client 1 : 10.0.2.13
Enter the lower port range: 65
Enter the higer port ranger: 70
port  65  is closed
port  66  is closed
port  67  is closed
port  68  is open
port  69  is closed
Server Responding To Client IP Address :
 ('10.0.2.13', 55054)
```