Screen Sharing program using Socket Programming

Course title: Computer Communication Networks
Course code: UE19EC301

Semester: V	Section: A
Amandeep Singh	PES1UG19EC034
Anshul Manapure	PES1UG19EC049

Aim:

To mirror the screen of one client on another's screen over a local network and to analyze the packets using wireshark.

Problem Statement:

Using socket programming, write a program to share your screen to another user.

Theory:

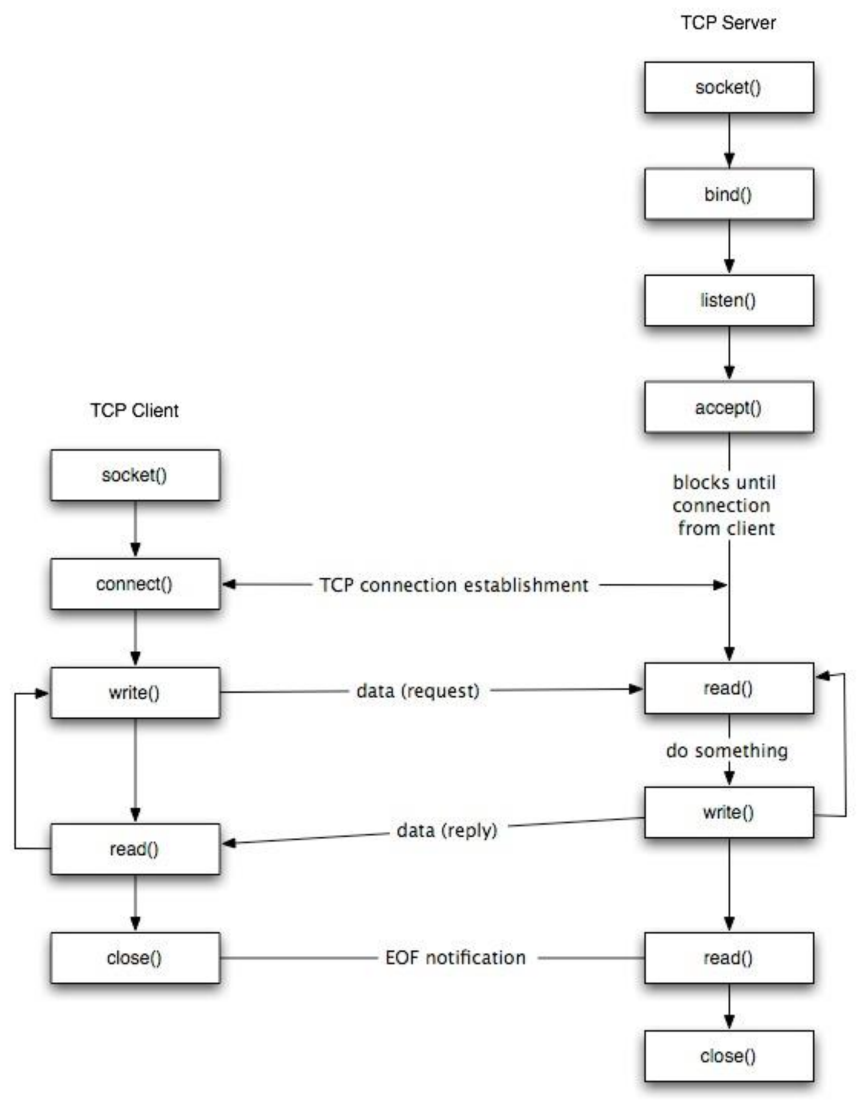
- Screen sharing is a peer-to-peer application as there is no specified server or client, both users can share their screen as required.
- The users first establish a TCP connection to ensure reliability while transferring information.
- The user sharing their screen continuously takes screenshots, compresses them and sends them to the other user.
- The receiver decompresses the data and displays the image in a new window.
- We use various python libraries like:
    1. Mss for capturing screenshots.
    2. Zlib for compressing and decompressing the image to be sent.
    3. Threading to ensure that multiple operations are performed as fast as possible.

4. Pygame to open a window and display the received screenshots.

Procedure:

- Run the program on the computer whose screen needs to be shared and enter the appropriate option
- Run the program on the computer where the screen shared is to be observed.
- Run the Wireshark packet capture in the background as the program runs.
- Observe the packets captured during the process.

Block Diagram:



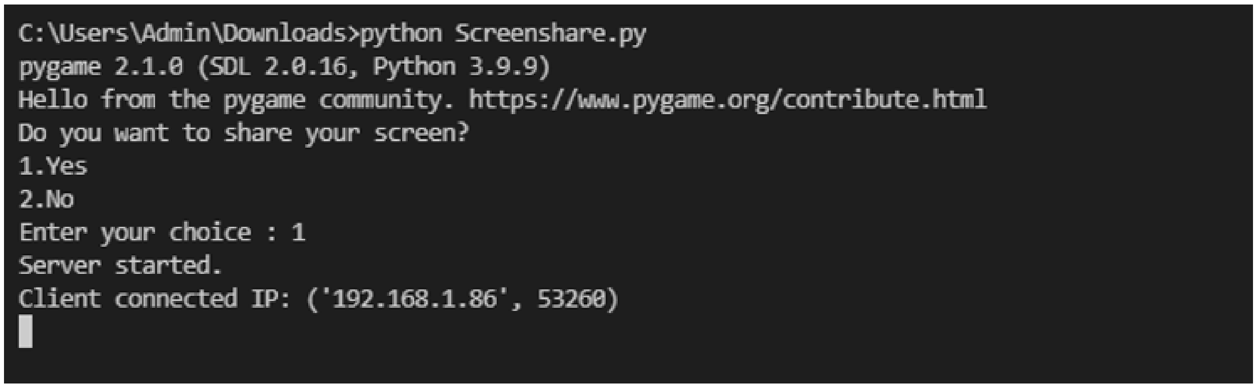
Working of a TCP connection(establishment and sharing of resources) as used in this project.

Code:

```
rom socket import socket
from threading import Thread
from zlib import compress
from mss import mss
from zlib import decompress
import pygame

WIDTH = 1366
HEIGHT = 768
host='192.168.1.82'
port=9000

ch=int(input('Do you want to share your screen?\n1.Yes\n2.No\nEnter your choice :
'))

if(ch==1):
    def retreive_screenshot(conn):
        with mss() as sct:
            # The region to capture
            rect = {'top': 0, 'left': 0, 'width': WIDTH, 'height': HEIGHT}

            while 'recording':
                # Capture the screen
                img = sct.grab(rect)
                # Tweak the compression level here (0-9)
                pixels = compress(img.rgb, 6)

                # Send the the pixels length
                size = len(pixels)
                size_len = (size.bit_length() + 7) // 8
                conn.send(bytes([size_len]))

                # Send the actual pixels length
                size_bytes = size.to_bytes(size_len, 'big')
                conn.send(size_bytes)

                # Send pixels
                conn.sendall(pixels)


    def server():
        sock = socket()
        sock.bind((host, port))
        try:
            sock.listen(5)
            print('Server started.')
            while 'connected':
                conn, addr = sock.accept()
                print('Client connected IP:', addr)
                thread = Thread(target=retreive_screenshot, args=(conn,))
                thread.start()
        finally:
            sock.close()

    server()

elif ch==2:
    def recvall(conn, length):
        """ Retreive all pixels. """

        buf = b''
        while len(buf) < length:
            data = conn.recv(length - len(buf))
            if not data:
                return data
            buf += data
        return buf


    def client():
        pygame.init()
        screen = pygame.display.set_mode((WIDTH, HEIGHT))
        clock = pygame.time.Clock()
        watching = True

        sock = socket()
        sock.connect((host, port))
        try:
            while watching:
                for event in pygame.event.get():
                    if event.type == pygame.QUIT:
                        watching = False
                        break

                # Retreive the size of the pixels, the pixels length and pixels
                size_len = int.from_bytes(sock.recv(1), byteorder='big')
                size = int.from_bytes(sock.recv(size_len), byteorder='big')
                pixels = decompress(recvall(sock, size))

                # Create the Surface from raw pixels
                img = pygame.image.fromstring(pixels, (WIDTH, HEIGHT), 'RGB')

                # Display the picture
                screen.blit(img, (0, 0))
                pygame.display.flip()
                clock.tick(60)
        finally:
            sock.close()

    client()

else:
    print('Invalid Choice.')
```

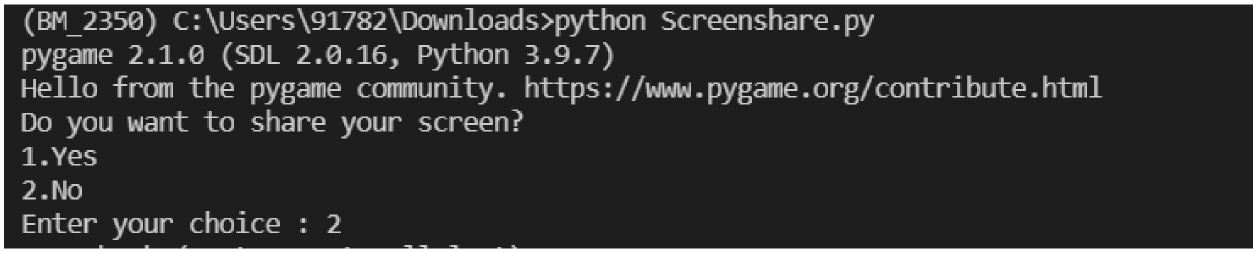
Output:

Server side terminal:



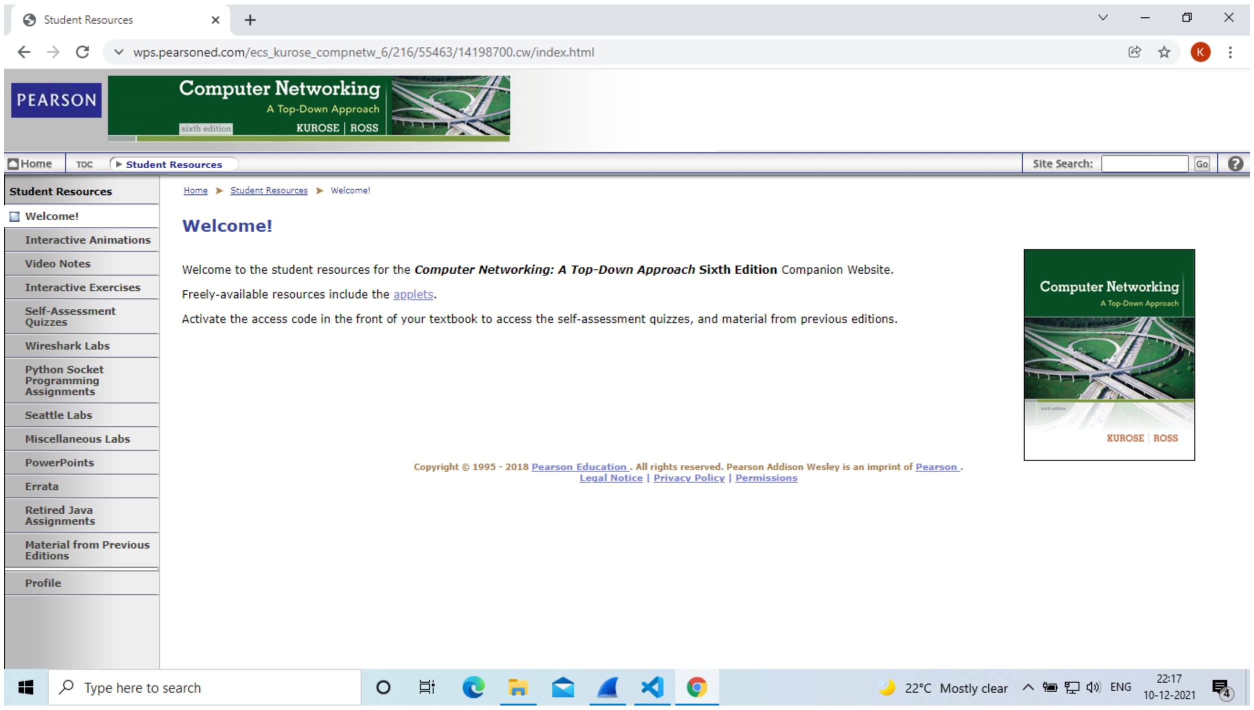
```
C:\Users\Admin\Downloads>python Screenshare.py
pygame 2.1.0 (SDL 2.0.16, Python 3.9.9)
Hello from the pygame community. https://www.pygame.org/contribute.html
Do you want to share your screen?
1.Yes
2.No
Enter your choice : 1
Server started.
Client connected IP: ('192.168.1.86', 53260)
```

Client Side terminal:



```
(BM_2350) C:\Users\91782\Downloads>python Screenshare.py
pygame 2.1.0 (SDL 2.0.16, Python 3.9.7)
Hello from the pygame community. https://www.pygame.org/contribute.html
Do you want to share your screen?
1.Yes
2.No
Enter your choice : 2
```

Server’s screen:



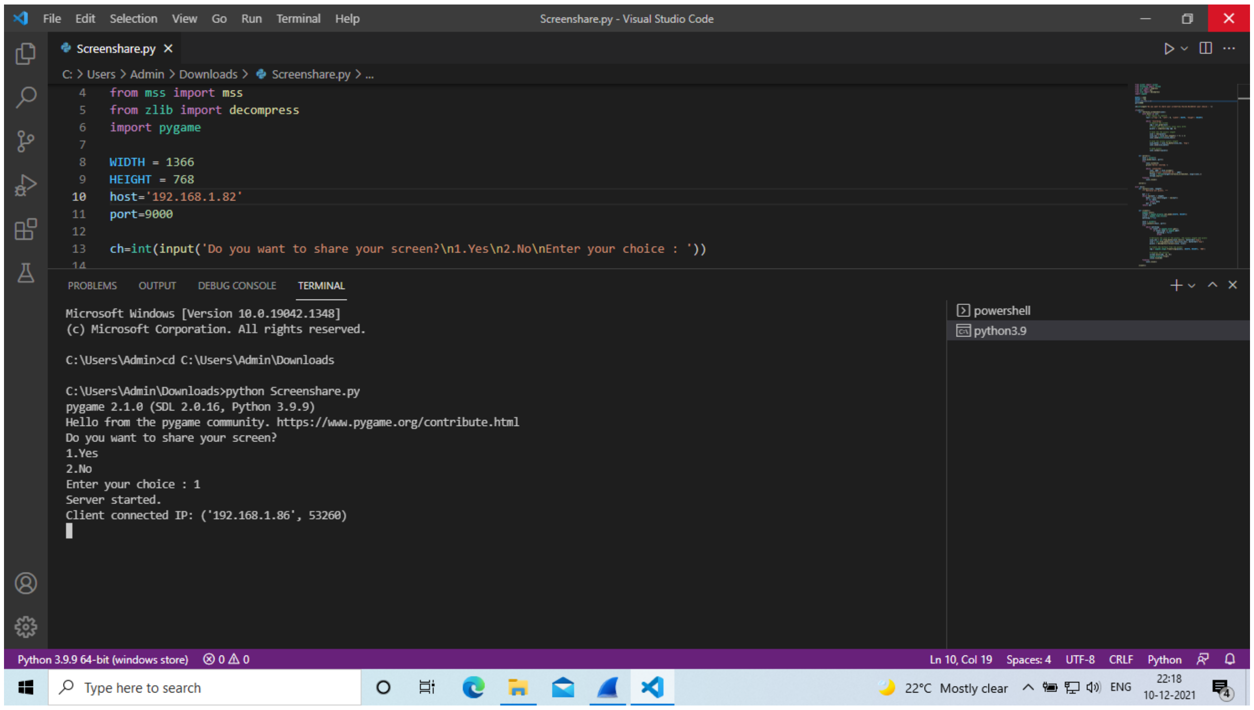
Welcome!

Welcome to the student resources for the ***Computer Networking: A Top-Down Approach* Sixth Edition** Companion Website.

Freely-available resources include the applets.

Activate the access code in the front of your textbook to access the self-assessment quizzes, and material from previous editions.

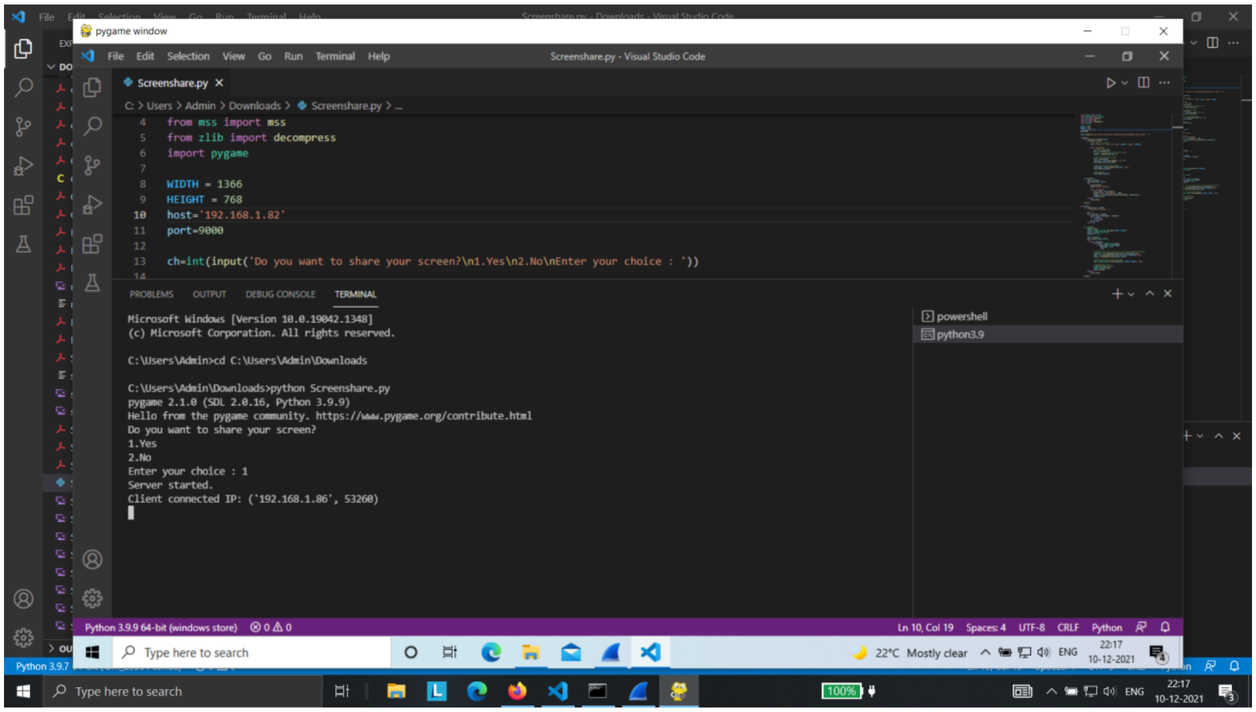
Copyright © 1995 - 2018 Pearson Education . All rights reserved. Pearson Addison Wesley is an imprint of Pearson .
Legal Notice | Privacy Policy | Permissions


File Edit Selection View Go Run Terminal Help
Screenshare.py - Visual Studio Code
Screenshare.py
C: > Users > Admin > Downloads > Screenshare.py > ...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
powershell
python3.9
Python 3.9.9 64-bit (windows store)
Ln 10, Col 19
Spaces: 4
UTF-8
CRLF
Python
Type here to search
22°C Mostly clear
ENG
22:18
10-12-2021

```python
from mss import mss
from zlib import decompress
import pygame

WIDTH = 1366
HEIGHT = 768
host='192.168.1.82'
port=9000

ch=int(input('Do you want to share your screen?\n1.Yes\n2.No\nEnter your choice : '))
```

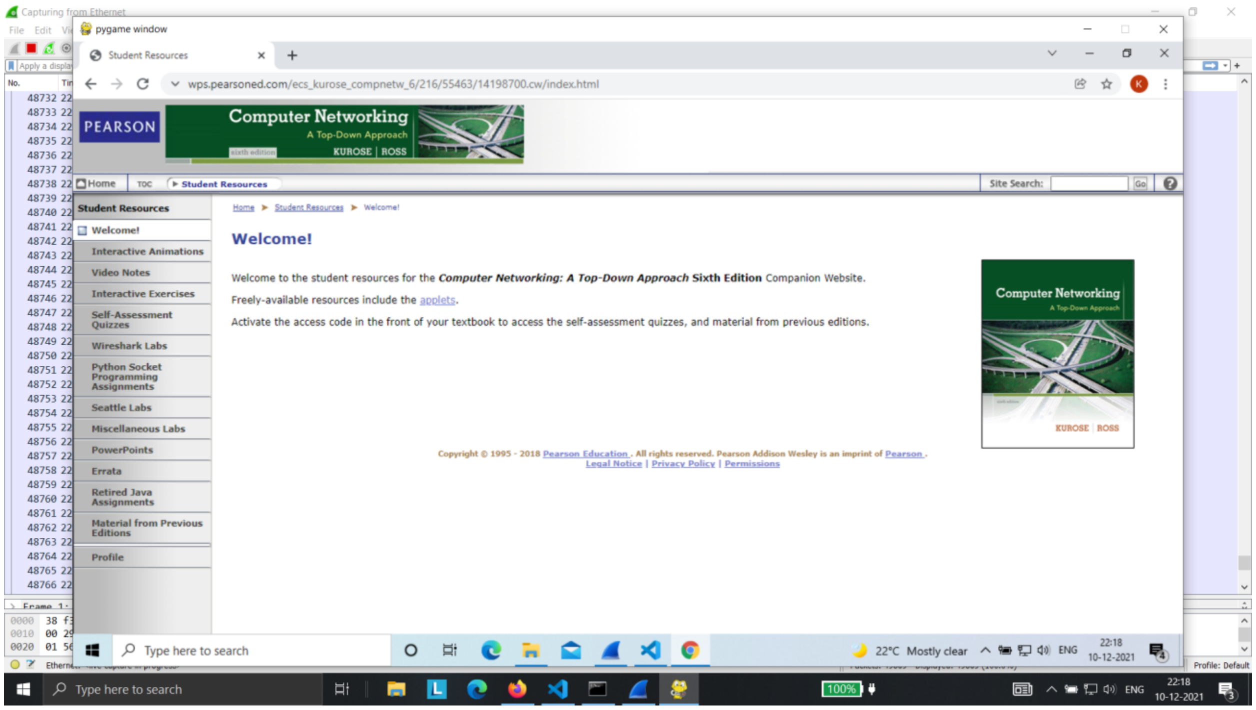
```
Microsoft Windows [Version 10.0.19042.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd C:\Users\Admin\Downloads

C:\Users\Admin\Downloads>python Screenshare.py
pygame 2.1.0 (SDL 2.0.16, Python 3.9.9)
Hello from the pygame community. https://www.pygame.org/contribute.html
Do you want to share your screen?
1.Yes
2.No
Enter your choice : 1
Server started.
Client connected IP: ('192.168.1.86', 53260)
```

Client's screen:



```
from mss import mss
from zlib import decompress
import pygame

WIDTH = 1366
HEIGHT = 768
host='192.168.1.82'
port=9000

ch=int(input('Do you want to share your screen?\n1.Yes\n2.No\nEnter your choice : '))
```

```
Microsoft Windows [Version 10.0.19042.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd C:\Users\Admin\Downloads

C:\Users\Admin\Downloads>python Screenshare.py
pygame 2.1.0 (SDL 2.0.16, Python 3.9.9)
Hello from the pygame community. https://www.pygame.org/contribute.html
Do you want to share your screen?
1.Yes
2.No
Enter your choice : 1
Server started.
Client connected IP: ('192.168.1.86', 53260)
```



wps.pearsoned.com/ecs_kurose_compnetw_6/216/55463/14198700.cw/index.html

Welcome!

Welcome to the student resources for the ***Computer Networking: A Top-Down Approach* Sixth Edition** Companion Website.

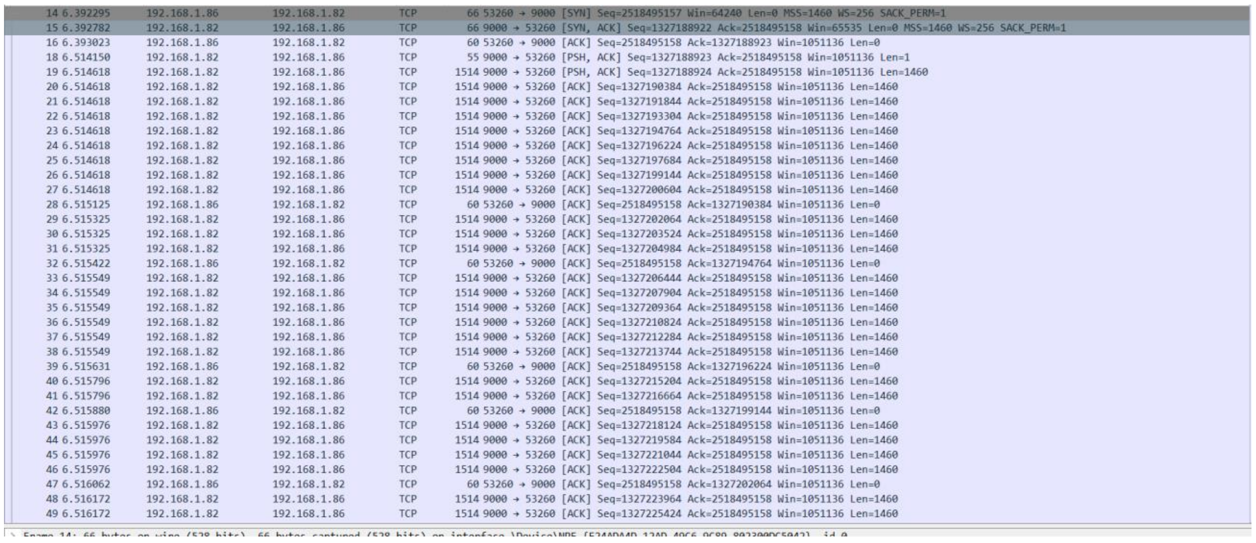
Freely-available resources include the applets.

Activate the access code in the front of your textbook to access the self-assessment quizzes, and material from previous editions.

Copyright © 1995 - 2018 Pearson Education . All rights reserved. Pearson Addison Wesley is an imprint of Pearson .
Legal Notice | Privacy Policy | Permissions

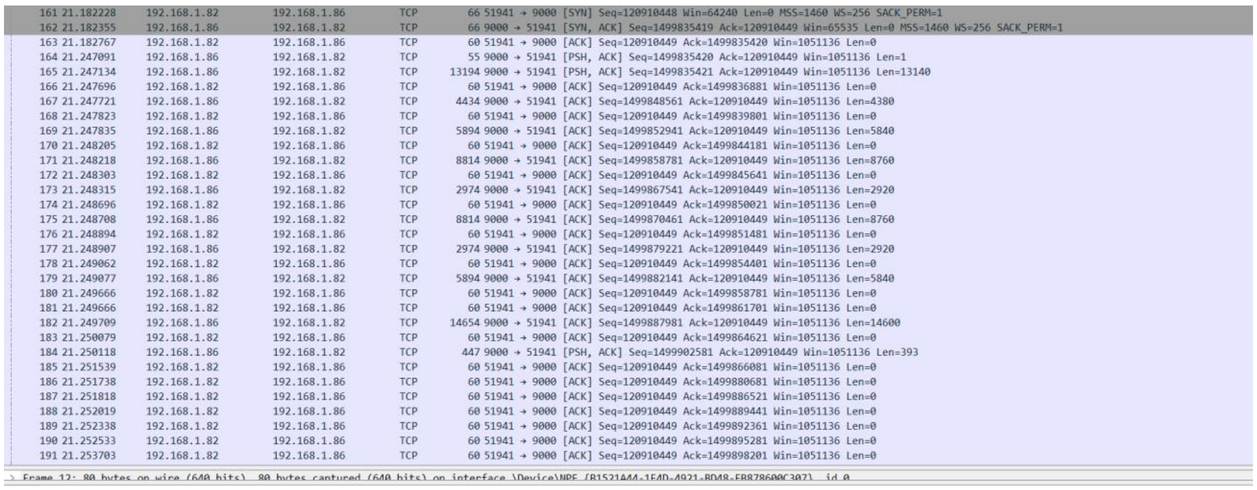
Wireshark Packets captured:

Server side:

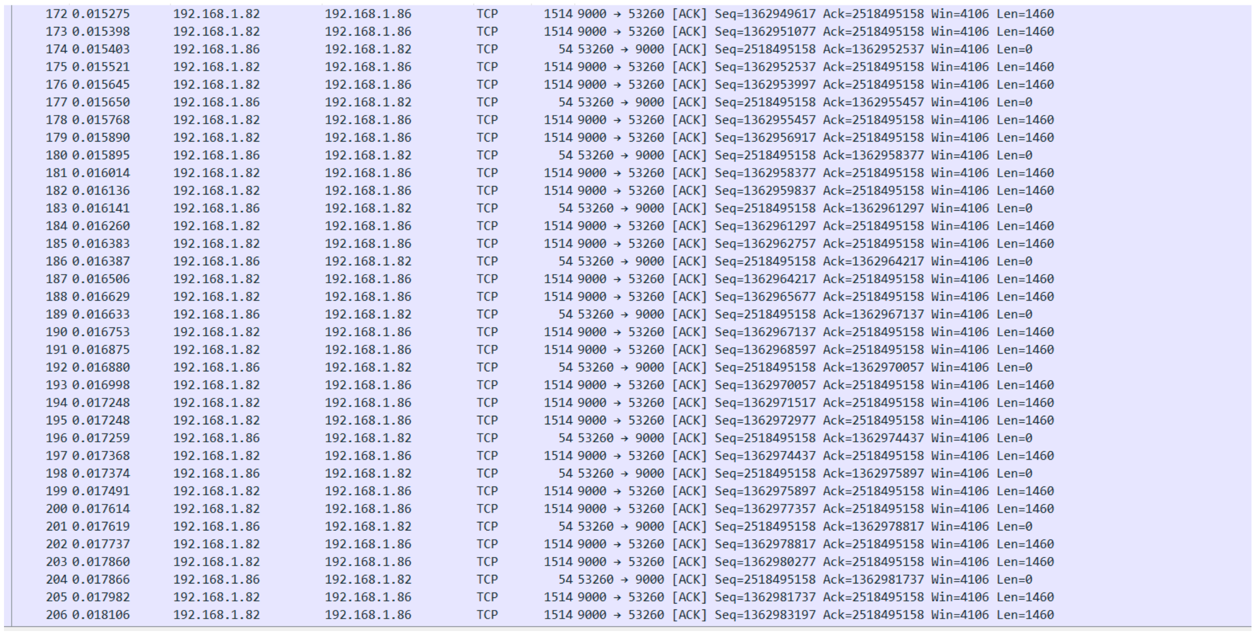


```
     14 6.392295     192.168.1.86      192.168.1.82      TCP      66 53260 → 9000 [SYN] Seq=2518495157 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
     15 6.392782     192.168.1.82      192.168.1.86      TCP      66 9000 → 53260 [SYN, ACK] Seq=1327188922 Ack=2518495158 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
     16 6.393023     192.168.1.86      192.168.1.82      TCP      60 53260 → 9000 [ACK] Seq=2518495158 Ack=1327188923 Win=1051136 Len=0
     18 6.514150     192.168.1.82      192.168.1.86      TCP      55 9000 → 53260 [PSH, ACK] Seq=1327188923 Ack=2518495158 Win=1051136 Len=1
     19 6.514618     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [PSH, ACK] Seq=1327188924 Ack=2518495158 Win=1051136 Len=1460
     20 6.514618     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327190384 Ack=2518495158 Win=1051136 Len=1460
     21 6.514618     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327191844 Ack=2518495158 Win=1051136 Len=1460
     22 6.514618     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327193304 Ack=2518495158 Win=1051136 Len=1460
     23 6.514618     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327194764 Ack=2518495158 Win=1051136 Len=1460
     24 6.514618     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327196224 Ack=2518495158 Win=1051136 Len=1460
     25 6.514618     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327197684 Ack=2518495158 Win=1051136 Len=1460
     26 6.514618     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327199144 Ack=2518495158 Win=1051136 Len=1460
     27 6.514618     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327200604 Ack=2518495158 Win=1051136 Len=1460
     28 6.515125     192.168.1.86      192.168.1.82      TCP      60 53260 → 9000 [ACK] Seq=2518495158 Ack=1327190384 Win=1051136 Len=0
     29 6.515325     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327202064 Ack=2518495158 Win=1051136 Len=1460
     30 6.515325     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327203524 Ack=2518495158 Win=1051136 Len=1460
     31 6.515325     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327204984 Ack=2518495158 Win=1051136 Len=1460
     32 6.515422     192.168.1.86      192.168.1.82      TCP      60 53260 → 9000 [ACK] Seq=2518495158 Ack=1327194764 Win=1051136 Len=0
     33 6.515549     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327206444 Ack=2518495158 Win=1051136 Len=1460
     34 6.515549     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327207904 Ack=2518495158 Win=1051136 Len=1460
     35 6.515549     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327209364 Ack=2518495158 Win=1051136 Len=1460
     36 6.515549     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327210824 Ack=2518495158 Win=1051136 Len=1460
     37 6.515549     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327212284 Ack=2518495158 Win=1051136 Len=1460
     38 6.515549     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327213744 Ack=2518495158 Win=1051136 Len=1460
     39 6.515631     192.168.1.86      192.168.1.82      TCP      60 53260 → 9000 [ACK] Seq=2518495158 Ack=1327196224 Win=1051136 Len=0
     40 6.515796     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327215204 Ack=2518495158 Win=1051136 Len=1460
     41 6.515796     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327216664 Ack=2518495158 Win=1051136 Len=1460
     42 6.515880     192.168.1.86      192.168.1.82      TCP      60 53260 → 9000 [ACK] Seq=2518495158 Ack=1327199144 Win=1051136 Len=0
     43 6.515976     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327218124 Ack=2518495158 Win=1051136 Len=1460
     44 6.515976     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327219584 Ack=2518495158 Win=1051136 Len=1460
     45 6.515976     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327221044 Ack=2518495158 Win=1051136 Len=1460
     46 6.515976     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327222504 Ack=2518495158 Win=1051136 Len=1460
     47 6.516062     192.168.1.86      192.168.1.82      TCP      60 53260 → 9000 [ACK] Seq=2518495158 Ack=1327202064 Win=1051136 Len=0
     48 6.516172     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327223964 Ack=2518495158 Win=1051136 Len=1460
     49 6.516172     192.168.1.82      192.168.1.86      TCP    1514 9000 → 53260 [ACK] Seq=1327225424 Ack=2518495158 Win=1051136 Len=1460
> Frame 14: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{E24ADA4D-12AD-49C6-9C89-802300DC5042}, id 0
```

Client side:



```
    161 21.182228    192.168.1.82      192.168.1.86      TCP      66 51941 → 9000 [SYN] Seq=120910448 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
    162 21.182355    192.168.1.86      192.168.1.82      TCP      66 9000 → 51941 [SYN, ACK] Seq=1499835419 Ack=120910449 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
    163 21.182767    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499835420 Win=1051136 Len=0
    164 21.247091    192.168.1.86      192.168.1.82      TCP      55 9000 → 51941 [PSH, ACK] Seq=1499835420 Ack=120910449 Win=1051136 Len=1
    165 21.247134    192.168.1.86      192.168.1.82      TCP   13194 9000 → 51941 [PSH, ACK] Seq=1499835421 Ack=120910449 Win=1051136 Len=13140
    166 21.247696    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499836881 Win=1051136 Len=0
    167 21.247721    192.168.1.86      192.168.1.82      TCP    4434 9000 → 51941 [ACK] Seq=1499848561 Ack=120910449 Win=1051136 Len=4380
    168 21.247823    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499839801 Win=1051136 Len=0
    169 21.247835    192.168.1.86      192.168.1.82      TCP    5894 9000 → 51941 [ACK] Seq=1499852941 Ack=120910449 Win=1051136 Len=5840
    170 21.248205    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499844181 Win=1051136 Len=0
    171 21.248218    192.168.1.86      192.168.1.82      TCP    8814 9000 → 51941 [ACK] Seq=1499858781 Ack=120910449 Win=1051136 Len=8760
    172 21.248303    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499845641 Win=1051136 Len=0
    173 21.248315    192.168.1.86      192.168.1.82      TCP    2974 9000 → 51941 [ACK] Seq=1499867541 Ack=120910449 Win=1051136 Len=2920
    174 21.248696    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499850021 Win=1051136 Len=0
    175 21.248708    192.168.1.86      192.168.1.82      TCP    8814 9000 → 51941 [ACK] Seq=1499870461 Ack=120910449 Win=1051136 Len=8760
    176 21.248894    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499851481 Win=1051136 Len=0
    177 21.248907    192.168.1.86      192.168.1.82      TCP    2974 9000 → 51941 [ACK] Seq=1499879221 Ack=120910449 Win=1051136 Len=2920
    178 21.249062    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499854401 Win=1051136 Len=0
    179 21.249077    192.168.1.86      192.168.1.82      TCP    5894 9000 → 51941 [ACK] Seq=1499882141 Ack=120910449 Win=1051136 Len=5840
    180 21.249666    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499858781 Win=1051136 Len=0
    181 21.249666    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499861701 Win=1051136 Len=0
    182 21.249709    192.168.1.86      192.168.1.82      TCP   14654 9000 → 51941 [ACK] Seq=1499887981 Ack=120910449 Win=1051136 Len=14600
    183 21.250079    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499864621 Win=1051136 Len=0
    184 21.250118    192.168.1.86      192.168.1.82      TCP     447 9000 → 51941 [PSH, ACK] Seq=1499902581 Ack=120910449 Win=1051136 Len=393
    185 21.251539    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499866081 Win=1051136 Len=0
    186 21.251738    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499880681 Win=1051136 Len=0
    187 21.251818    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499886521 Win=1051136 Len=0
    188 21.252019    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499889441 Win=1051136 Len=0
    189 21.252338    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499892361 Win=1051136 Len=0
    190 21.252533    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499895281 Win=1051136 Len=0
    191 21.253703    192.168.1.82      192.168.1.86      TCP      60 51941 → 9000 [ACK] Seq=120910449 Ack=1499898201 Win=1051136 Len=0
> Frame 12: 80 bytes on wire (640 bits), 80 bytes captured (640 bits) on interface \Device\NPF_{B1521A44-1E4D-4921-BD48-EB878600C307}, id 0
```



```
    172 0.015275    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362949617 Ack=2518495158 Win=4106 Len=1460
    173 0.015398    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362951077 Ack=2518495158 Win=4106 Len=1460
    174 0.015403    192.168.1.86    192.168.1.82    TCP      54 53260 → 9000 [ACK] Seq=2518495158 Ack=1362952537 Win=4106 Len=0
    175 0.015521    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362952537 Ack=2518495158 Win=4106 Len=1460
    176 0.015645    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362953997 Ack=2518495158 Win=4106 Len=1460
    177 0.015650    192.168.1.86    192.168.1.82    TCP      54 53260 → 9000 [ACK] Seq=2518495158 Ack=1362955457 Win=4106 Len=0
    178 0.015768    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362955457 Ack=2518495158 Win=4106 Len=1460
    179 0.015890    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362956917 Ack=2518495158 Win=4106 Len=1460
    180 0.015895    192.168.1.86    192.168.1.82    TCP      54 53260 → 9000 [ACK] Seq=2518495158 Ack=1362958377 Win=4106 Len=0
    181 0.016014    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362958377 Ack=2518495158 Win=4106 Len=1460
    182 0.016136    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362959837 Ack=2518495158 Win=4106 Len=1460
    183 0.016141    192.168.1.86    192.168.1.82    TCP      54 53260 → 9000 [ACK] Seq=2518495158 Ack=1362961297 Win=4106 Len=0
    184 0.016260    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362961297 Ack=2518495158 Win=4106 Len=1460
    185 0.016383    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362962757 Ack=2518495158 Win=4106 Len=1460
    186 0.016387    192.168.1.86    192.168.1.82    TCP      54 53260 → 9000 [ACK] Seq=2518495158 Ack=1362964217 Win=4106 Len=0
    187 0.016506    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362964217 Ack=2518495158 Win=4106 Len=1460
    188 0.016629    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362965677 Ack=2518495158 Win=4106 Len=1460
    189 0.016633    192.168.1.86    192.168.1.82    TCP      54 53260 → 9000 [ACK] Seq=2518495158 Ack=1362967137 Win=4106 Len=0
    190 0.016753    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362967137 Ack=2518495158 Win=4106 Len=1460
    191 0.016875    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362968597 Ack=2518495158 Win=4106 Len=1460
    192 0.016880    192.168.1.86    192.168.1.82    TCP      54 53260 → 9000 [ACK] Seq=2518495158 Ack=1362970057 Win=4106 Len=0
    193 0.016998    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362970057 Ack=2518495158 Win=4106 Len=1460
    194 0.017248    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362971517 Ack=2518495158 Win=4106 Len=1460
    195 0.017248    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362972977 Ack=2518495158 Win=4106 Len=1460
    196 0.017259    192.168.1.86    192.168.1.82    TCP      54 53260 → 9000 [ACK] Seq=2518495158 Ack=1362974437 Win=4106 Len=0
    197 0.017368    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362974437 Ack=2518495158 Win=4106 Len=1460
    198 0.017374    192.168.1.86    192.168.1.82    TCP      54 53260 → 9000 [ACK] Seq=2518495158 Ack=1362975897 Win=4106 Len=0
    199 0.017491    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362975897 Ack=2518495158 Win=4106 Len=1460
    200 0.017614    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362977357 Ack=2518495158 Win=4106 Len=1460
    201 0.017619    192.168.1.86    192.168.1.82    TCP      54 53260 → 9000 [ACK] Seq=2518495158 Ack=1362978817 Win=4106 Len=0
    202 0.017737    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362978817 Ack=2518495158 Win=4106 Len=1460
    203 0.017860    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362980277 Ack=2518495158 Win=4106 Len=1460
    204 0.017866    192.168.1.86    192.168.1.82    TCP      54 53260 → 9000 [ACK] Seq=2518495158 Ack=1362981737 Win=4106 Len=0
    205 0.017982    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362981737 Ack=2518495158 Win=4106 Len=1460
    206 0.018106    192.168.1.82    192.168.1.86    TCP    1514 9000 → 53260 [ACK] Seq=1362983197 Ack=2518495158 Win=4106 Len=1460
```

Result:

This program and outputs provide enough proof that the process of screensharing is possible using socket programming while also being reliable, as seen in the screenshots, due to the use of the transport layer protocol, TCP.