Aleena Anna Plessey
21BCE5767

RTE Socket Programming

Flow Control Mechanism

Submitted by Aleena Anna Plessey

Reg no: 21BCE5767

Course Program: B.Tech

Course code: BCSE308L

Course Title: Computer Networks

Batch:2021-25

Submitted to: Dr. Swaminathan A

Submitted on 21/06/2023

Problem Statement:

9.1 Write a program to implement the Stop and wait with ARQ using socket programing.

9.2 Write a program to implement the Go-back-N with ARQ using socket programing.

9.3 Write a program to implement the Selective repeat with using socket programing.

Aim and Objective:

To implement flow control mechanisms like stop and wait with ARQ,Go back N withARQ and Selective repeat using socket programming in python,

Solutions:

9.1 Stop and wait with ARQ using socket programing.

Code:

Client side:

```python
import socket
SERVER_ADDRESS = 'localhost'
SERVER_PORT = 12345
BUFFER_SIZE = 1024

client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

while True:
    message = input("Enter a message to send: ")
    client_socket.sendto(message.encode(), (SERVER_ADDRESS, SERVER_PORT))
    ack, server_address = client_socket.recvfrom(BUFFER_SIZE)
    print("Received acknowledgement:", ack.decode())

```

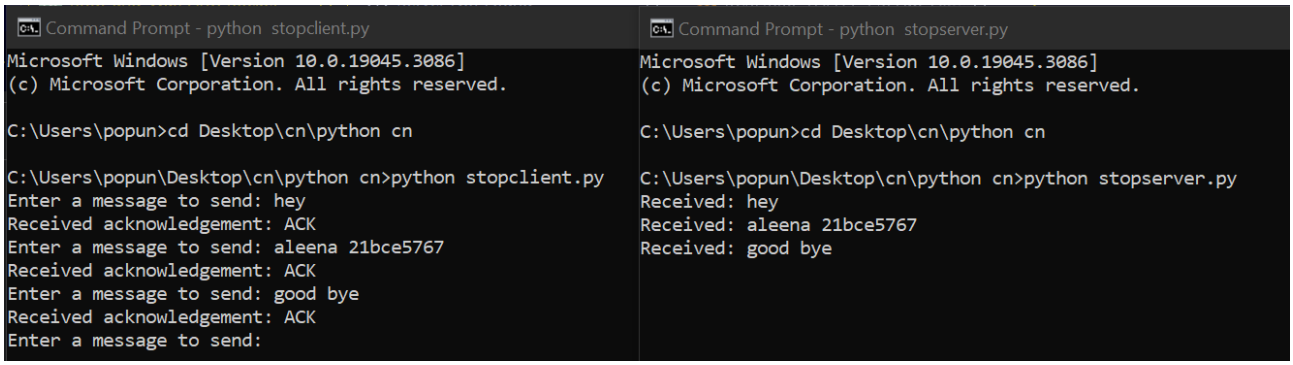
Server side:

```python
import socket
SERVER_ADDRESS = 'localhost'
SERVER_PORT = 12345
BUFFER_SIZE = 1024
server_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
server_socket.bind((SERVER_ADDRESS, SERVER_PORT))

while True:
    data, client_address = server_socket.recvfrom(BUFFER_SIZE)
    print("Received:", data.decode())
    ack = "ACK"
    server_socket.sendto(ack.encode(), client_address)

```

1)Run the server-side code first to start the server.

2)Run the client-side server and input the message the output will be acknowledged.

Output:



9.2 Go-back-N with ARQ using socket programing.

Code:

Client:

```python
import socket
SERVER_ADDRESS = 'localhost'
SERVER_PORT = 12345
BUFFER_SIZE = 1024
WINDOW_SIZE = 4
client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
messages = []
num_messages = int(input("Enter the number of messages: "))
for _ in range(num_messages):
    message = input("Enter a message to send: ")
    messages.append(message)

for message in messages:
    packets = [message[i:i+BUFFER_SIZE] for i in range(0, len(message), BUFFER_SIZE)]
    expected_ack = 0

    while expected_ack < len(packets):
        for i in range(expected_ack, min(expected_ack + WINDOW_SIZE, len(packets))):
            client_socket.sendto(packets[i].encode(), (SERVER_ADDRESS, SERVER_PORT))
            print("Sent:", packets[i])
        while True:
            try:
                client_socket.settimeout(1.0)
                ack, server_address = client_socket.recvfrom(BUFFER_SIZE)
                expected_ack = int(ack.decode()) + 1
                print("Received acknowledgement:", ack.decode())
                break
            except socket.timeout:
                print("Timeout occurred. Resending packets...")
                break
```

Server:

```python
import socket
SERVER_ADDRESS = 'localhost'
SERVER_PORT = 12345
BUFFER_SIZE = 1024
WINDOW_SIZE = 4
server_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
server_socket.bind((SERVER_ADDRESS, SERVER_PORT))

while True:
    data, client_address = server_socket.recvfrom(BUFFER_SIZE)
    print("Received:", data.decode())
    packets = [data[i:i+BUFFER_SIZE] for i in range(0, len(data), BUFFER_SIZE)]
    ack = len(packets) - 1
    server_socket.sendto(str(ack).encode(), client_address)
```

1)Run the server-side code first to start the server.

2)Run the client-side server and send the first message. It will be reflected and the server side will get the message.

3)Then the server side can send the message which will be received in the client side.

Output:

```
Microsoft Windows [Version 10.0.19045.3086]
(c) Microsoft Corporation. All rights reserved.

C:\Users\popun>cd Desktop\cn\python cn

C:\Users\popun\Desktop\cn\python cn>python stopserver.py
Received: hey
Received: aleena here
Received: 21bce5767
```

```
Command Prompt
Microsoft Windows [Version 10.0.19045.3086]
(c) Microsoft Corporation. All rights reserved.

C:\Users\popun>cd Desktop\cn\python cn

C:\Users\popun\Desktop\cn\python cn>python stopclient.py
Enter the number of messages: 3
Enter a message to send: hey
Enter a message to send: aleena here
Enter a message to send: 21bce5767
Sent: hey
Received acknowledgement: 0
Sent: aleena here
Received acknowledgement: 0
Sent: 21bce5767
Received acknowledgement: 0

C:\Users\popun\Desktop\cn\python cn>
```

9.3 Selective repeat with using socket programing.

Code:

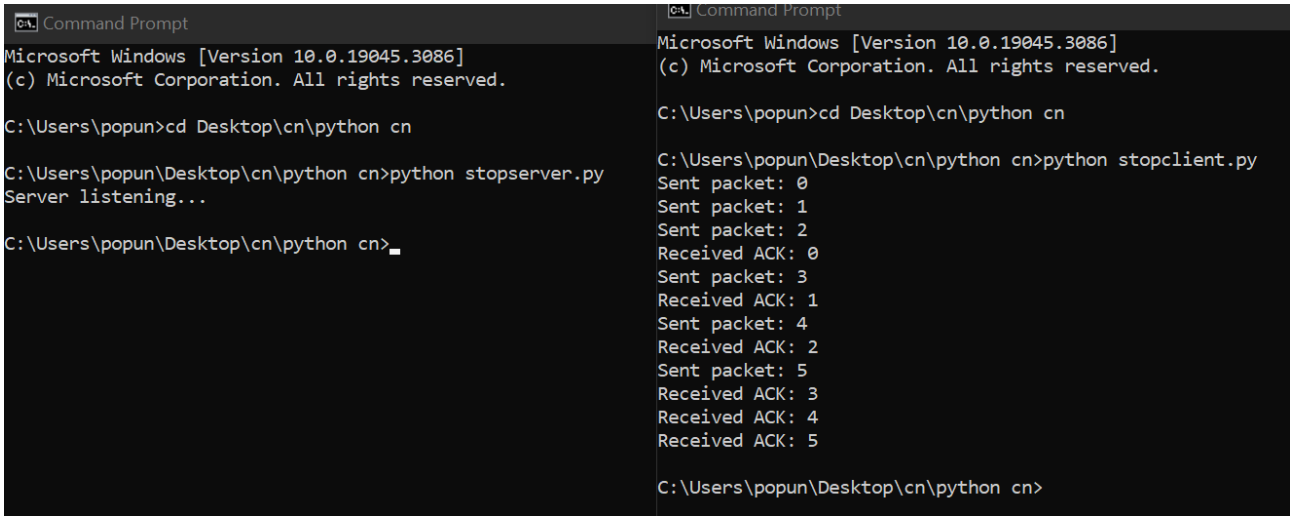
Server:

```python
import socket
server_ip = '127.0.0.1'
server_port = 12345
server_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
server_socket.bind((server_ip, server_port))
print("Server listening...")
window_size = 3
while True:
    data, addr = server_socket.recvfrom(1024)
    sequence_number = int(data.decode())
    if sequence_number == -1:
        break
    ack = str(sequence_number)
    server_socket.sendto(ack.encode(), addr)
server_socket.close()

```

Client:

```python
import socket
import time
server_ip = '127.0.0.1'
server_port = 12345
client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
window_size = 3
packets = [0, 1, 2, 3, 4, 5]
base = 0
next_seq_num = 0

while base < len(packets):
    while next_seq_num < base + window_size and next_seq_num < len(packets):
        packet = str(packets[next_seq_num])
        client_socket.sendto(packet.encode(), (server_ip, server_port))
        print("Sent packet:", packet)
        next_seq_num += 1
    while True:
        try:
            client_socket.settimeout(2)
            ack, addr = client_socket.recvfrom(1024)
            ack = int(ack.decode())
            print("Received ACK:", ack)
            if ack >= base:
                base = ack + 1
                break

        except socket.timeout:
            print("Timeout occurred. Resending packets...")
            next_seq_num = base
            break
client_socket.sendto(str(-1).encode(), (server_ip, server_port))
client_socket.close()
```

Output:



Result:

We were able to successfully implement commonly used flow control mechanisms with ARQ in python programming language.