CPSC 471-01 Project 2

Team: Luc Dang (solo)

Part 1 – UDP Pinger

Instructions:

1. On a UNIX system open the system terminal and navigate to the appropriate directory containing the *udppingserver.py* and *ldp21c.py* files.
2. In the terminal, run the server program first with the command "*python3 udppingserver.py".*
3. Open a new tab in the terminal, or a separate terminal window, and run the client program with the command *"python3 ldp21c.py NUM_PINGS"* where NUM_PINGS is the desired number of pings.

Screen captures:

a. A sequence consisting of 10 pings:

```
student@tuffix-vm:~/Desktop/p2$ python3 ldp21c.py 10
Ping 1 :  seq 1 19:48:40 , RTT = 0.0949 ms
Ping 2 :  Request timed out
Ping 3 :  seq 3 19:48:41 , RTT = 0.1175 ms
Ping 4 :  seq 4 19:48:41 , RTT = 0.0317 ms
Ping 5 :  Request timed out
Ping 6 :  Request timed out
Ping 7 :  seq 7 19:48:43 , RTT = 0.1402 ms
Ping 8 :  seq 8 19:48:43 , RTT = 0.0343 ms
Ping 9 :  seq 9 19:48:43 , RTT = 0.0317 ms
Ping 10 :  seq 10 19:48:43 , RTT = 0.0284 ms

--- SUMMARY ---
Minimum RTT:  0.0284 ms
Maximum RTT:  0.1402 ms
Average RTT:  0.0684 ms
Number of packets received:  7
Number of packets lost:  3
Packet loss percentage:  30.0 %
```

b. A sequence consisting of 15 pings:

```
student@tuffix-vm:~/Desktop/p2$ python3 ldp21c.py 15
Ping 1 :  seq 1 19:52:36 , RTT = 3.4149 ms
Ping 2 :  Request timed out
Ping 3 :  seq 3 19:52:37 , RTT = 0.1359 ms
Ping 4 :  seq 4 19:52:37 , RTT = 0.0329 ms
Ping 5 :  seq 5 19:52:37 , RTT = 0.0288 ms
Ping 6 :  seq 6 19:52:37 , RTT = 0.0272 ms
Ping 7 :  seq 7 19:52:37 , RTT = 0.0279 ms
Ping 8 :  Request timed out
Ping 9 :  Request timed out
Ping 10 :  seq 10 19:52:39 , RTT = 0.1318 ms
Ping 11 :  seq 11 19:52:39 , RTT = 0.031 ms
Ping 12 :  Request timed out
Ping 13 :  seq 13 19:52:40 , RTT = 0.1276 ms
Ping 14 :  Request timed out
Ping 15 :  seq 15 19:52:41 , RTT = 0.1299 ms

--- SUMMARY ---
Minimum RTT:  0.0272 ms
Maximum RTT:  3.4149 ms
Average RTT:  0.4088 ms
Number of packets received:  10
Number of packets lost:  5
Packet loss percentage:  33.33 %
```

Python code listing:

```
# ldp21c.py
import time
import sys
from socket import *

# UDP socket
clientSocket = socket(AF_INET, SOCK_DGRAM)
clientSocket.settimeout(1)
serv_addr = ('', 45678)

RTTarr = []
try:
    numPings = int(sys.argv[1])
except IndexError:
    print("\nIncorrect usage. Use: 'python3 ldp21c.py NUM_PINGS' where
NUM_PINGS is the desired number of pings.\n")

for x in range(numPings):
    sendTime = time.time()
    message = ('seq ' + str(x+1) + " " +
str(time.strftime("%H:%M:%S"))).encode()
    clientSocket.sendto(message, serv_addr)

    try:
        data, server = clientSocket.recvfrom(1024)
        recvTime = time.time()
        rtt = (recvTime - sendTime) * 1000
        RTTarr.append(rtt)
        print("Ping", str(x+1) ,": ", data.decode(), ", RTT =",round(rtt, 4),
"ms")

    except timeout:
        print("Ping", str(x+1) ,": ", "Request timed out")

print()
print("--- SUMMARY ---")
print("Minimum RTT: ", round(min(RTTarr), 4), "ms")
print("Maximum RTT: ", round(max(RTTarr), 4), "ms")
print("Average RTT: ", round((sum(RTTarr)/len(RTTarr)), 4), "ms")
print("Number of packets received: ", len(RTTarr))
print("Number of packets lost: ", numPings - len(RTTarr))
print("Packet loss percentage: ", round( ((numPings -
len(RTTarr))/numPings*100), 2), "%")
```

Part 2 – UDP Heartbeat Monitor

Instructions:

1. On a UNIX system open the system terminal and navigate to the appropriate directory containing the *ldp22s.py* and *ldp22c.py* files.
2. In the terminal, run the server program first with the command "*python3 ldp22s.py".*
3. Open a new tab in the terminal, or a separate terminal window, and run the client program with the command *"python3 ldp22c.py NUM_PINGS"* where NUM_PINGS is the desired number of pings.

Screen captures:

a. Client sending heartbeat pings every 5 seconds:

```
student@tuffix-vm:~/Desktop/p2$ python3 ldp22c.py 10
heartbeat pulse 1
heartbeat pulse 2
heartbeat pulse 3
heartbeat pulse 4
heartbeat pulse 5
heartbeat pulse 6
heartbeat pulse 7
heartbeat pulse 8
heartbeat pulse 9
heartbeat pulse 10
```

b. Server printing received heartbeat pings and time interval:

```
student@tuffix-vm:~/Desktop/p2$ python3 ldp22s.py
Server received heartbeat pulse 1. Pulse interval was 1.4 seconds.
Server received heartbeat pulse 2. Pulse interval was 5.0 seconds.
Server received heartbeat pulse 3. Pulse interval was 5.0 seconds.
Server received heartbeat pulse 4. Pulse interval was 5.0 seconds.
Server received heartbeat pulse 5. Pulse interval was 5.0 seconds.
Server received heartbeat pulse 6. Pulse interval was 5.0 seconds.
Server received heartbeat pulse 7. Pulse interval was 5.0 seconds.
Server received heartbeat pulse 8. Pulse interval was 5.0 seconds.
Server received heartbeat pulse 9. Pulse interval was 5.0 seconds.
Server received heartbeat pulse 10. Pulse interval was 5.0 seconds.
No pulse after 10 seconds. Server quits.
Server stops.
```

c. Server detects absence of client heartbeat and quits:

```
No pulse after 10 seconds. Server quits.
Server stops.
```

Python code listing:

a. Client program:

```
# ldp22c.py
import time
import sys
from socket import *

# UDP socket
clientSocket = socket(AF_INET, SOCK_DGRAM)
serv_addr = ('', 45678)

RTTarr = []
try:
    numPings = int(sys.argv[1])
except IndexError:
    print("\nIncorrect usage. Use: 'python3 ldp22c.py NUM_PINGS' where
NUM_PINGS is the desired number of pings.\n")

for x in range(numPings):
    message = ("heartbeat pulse " + str(x+1)).encode()
    clientSocket.sendto(message, serv_addr)
    print(message.decode())
    time.sleep(5)
```

b. Server program:

```
# ldp22s.py
import time
import sys
from socket import *

# Create a UDP socket
serverSocket = socket(AF_INET, SOCK_DGRAM)
# timeout after 10 seconds
serverSocket.settimeout(10)
# Assign IP address and port number to socket
serverSocket.bind(('', 45678))

interval = 0

while True:
    prevTime = time.time()
    try:
        message, address = serverSocket.recvfrom(1024)
        recvTime = time.time()
        interval = recvTime - prevTime
        print("Server received " + message.decode() + ". Pulse interval was "
+ str(round(interval, 1)) + " seconds.")

    except timeout:
        print("No pulse after 10 seconds. Server quits.\nServer stops.")
        serverSocket.close()
        sys.exit()
```