**Packet Generation Scripts for TCP with Variable Size Packets**

**Server**

import socket

import sys

import traceback

from threading import Thread

packet\_count = 0

def main():

start\_server()

def start\_server():

host = "11.0.0.5"

port = 8220 # arbitrary non-privileged port

soc = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

soc.setsockopt(socket.SOL\_SOCKET, socket.SO\_REUSEADDR, 1) # SO\_REUSEADDR flag tells the kernel to reuse a local socket in TIME\_WAIT state, without waiting for its natural timeout to expire

print("Socket created")

try:

soc.bind((host, port))

except:

print("Bind failed. Error : " + str(sys.exc\_info()))

sys.exit()

soc.listen(5) # queue up to 5 requests

print("Socket now listening")

# infinite loop- do not reset for every requests

while True:

connection, address = soc.accept()

ip, port = str(address[0]), str(address[1])

print("Connected with " + ip + ":" + port)

try:

Thread(target=client\_thread, args=(connection, ip, port)).start()

except:

print("Thread did not start.")

traceback.print\_exc()

soc.close()

def client\_thread(connection, ip, port, max\_buffer\_size = 16384):

global packet\_count

is\_active = True

while is\_active:

client\_input = receive\_input(connection, max\_buffer\_size)

packet\_count += 1

if "--quit--" in client\_input:

#print("Client is requesting to quit")

connection.close()

#print("Connection " + ip + ":" + port + " closed")

packet\_count -= 1

print("Packet Count = "+str(packet\_count))

is\_active = False

else:

print(client\_input)

connection.sendall("-".encode("utf8"))

def receive\_input(connection, max\_buffer\_size):

client\_input = connection.recv(max\_buffer\_size)

client\_input\_size = sys.getsizeof(client\_input)

if client\_input\_size > max\_buffer\_size:

print("The input size is greater than expected {}".format(client\_input\_size))

decoded\_input = client\_input.decode("utf8").rstrip() # decode and strip end of line

result = process\_input(decoded\_input)

return result

def process\_input(input\_str):

#print("Processing the input received from client")

return str(input\_str)

if \_\_name\_\_ == "\_\_main\_\_":

main()

Client

import socket

import sys

import random

from time import time

def main():

soc = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

host = "11.0.0.5"

port = 8220

randomClass = ["00","01","10","11"]

priorityTime = [0,0,0]

variable\_size = [2048,4096,8192,16384]

try:

soc.connect((host, port))

except:

print("Connection error")

sys.exit()

#print("Enter 'quit' to exit")

start\_time = time()

for i in range(10000):

start\_time\_packet = time()

randomIndex = random.randint(0,3)

message = randomClass[randomIndex] + " Some Random Data "

soc.sendall(message.encode("utf8"))

if soc.recv(variable\_size[randomIndex]).decode("utf8") == "-":

pass # null operation

end\_time\_packet = time()

elapsed\_time\_packet = end\_time\_packet - start\_time\_packet

hours,rest = divmod(elapsed\_time\_packet,3600)

minutes,seconds = divmod(rest,60)

seconds += (minutes\*60)

if randomClass[randomIndex] == "00":

priorityTime[0] += seconds

elif randomClass[randomIndex] == "10" or randomClass[randomIndex] == "01" :

priorityTime[1] += seconds

elif randomClass[randomIndex] == "11":

priorityTime[2] += seconds

soc.send(b'--quit--')

end\_time = time()

elapsed\_time = end\_time - start\_time

hours,rest = divmod(elapsed\_time,3600)

minutes,seconds = divmod(rest,60)

print(str(minutes)+' - '+str(seconds))

print()

print()

print("High: {0:.10f}".format(priorityTime[2]))

print("Medium: {0:.10f}".format(priorityTime[1]))

print("Low: {0:.10f}".format(priorityTime[0]))

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Packet Generation Scripts for UDP with Variable Size Packets**

**Server**

import socket

UDP\_IP = "11.0.0.5"

UDP\_PORT = 8220

packet\_count = 0

sock = socket.socket(socket.AF\_INET, # Internet

socket.SOCK\_DGRAM) # UDP

sock.bind((UDP\_IP, UDP\_PORT))

while True:

data, addr = sock.recvfrom(16384) # buffer size is 1024 bytes

#print "received message:", data

packet\_count+=1

if data == "--quit--":

packet\_count-=1

print("Packet Count = "+str(packet\_count))

print("Completed")

**Client**

import socket

import sys

import random

from time import time

def main():

soc = socket.socket(socket.AF\_INET, # Internet

socket.SOCK\_DGRAM) # UDP

UDP\_IP = "11.0.0.5"

UDP\_PORT = 8220

randomClass = ["11","01","10","00"]

priorityTime = [0,0,0]

variable\_size = [2048,4096,8192,16384]

Randomcharacters = ['a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z']

random\_size = random.randint(0,3)

text = ''

start\_time = time()

for i in range(40000):

text = ''

start\_time\_packet = time()

randomIndex = random.randint(0,3)

#for j in range(variable\_size[randomIndex]):

# text += Randomcharacters[random.randint(0,25)]

#message = randomClass[randomIndex] + text + '\n'

message = randomClass[randomIndex] + "Some Random Text"

soc.sendto(message, (UDP\_IP, UDP\_PORT))

#if soc.recvfrom(variable\_size[randomIndex]).decode("utf8") == "-":

# pass # null operation

end\_time\_packet = time()

elapsed\_time\_packet = end\_time\_packet - start\_time\_packet

hours,rest = divmod(elapsed\_time\_packet,3600)

minutes,seconds = divmod(rest,60)

seconds += (minutes\*60)

if randomClass[randomIndex] == "00":

priorityTime[0] += seconds

elif randomClass[randomIndex] == "10" or randomClass[randomIndex] == "01" :

priorityTime[1] += seconds

elif randomClass[randomIndex] == "11":

priorityTime[2] += seconds

soc.sendto("--quit--", (UDP\_IP, UDP\_PORT))

end\_time = time()

elapsed\_time = end\_time - start\_time

hours,rest = divmod(elapsed\_time,3600)

minutes,seconds = divmod(rest,60)

print(str(minutes)+' - '+str(seconds))

print()

print()

print("High: {0:.10f}".format(priorityTime[2]))

print("Medium: {0:.10f}".format(priorityTime[1]))

print("Low: {0:.10f}".format(priorityTime[0]))

if \_\_name\_\_ == "\_\_main\_\_":

main()