import hashlib

import os

import threading

from random import random

import utilesFiles

import Objetos

import socket

import time

from os import walk

ipEsteEquipo = ""

equipos = []

misArchivos = []

listaArchivos = []

def getListArchivos():

strRetorno = ""

for arc in listaArchivos:

strRetorno = strRetorno + "<" + str(arc.idArchivo) + "><" + str(arc.largo) + "><" + arc.nombre + ">\r\n"

return strRetorno

def escucharAnuncios():

client = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM, socket.IPPROTO\_UDP) # UDP

# Enable broadcasting mode

client.setsockopt(socket.SOL\_SOCKET, socket.SO\_BROADCAST, 1)

client.bind(("", 2020))

while True:

data, addr = client.recvfrom(1024)

if ipEsteEquipo != addr[0]:

print(" UDP Recibido en IP : " + ipEsteEquipo + " - origen : " + addr[0] )

equipo = Objetos.Equipo("0")

for eq in equipos:

if eq.ip == addr[0]:

equipo = eq

if equipo.ip == '0':

equipo = Objetos.Equipo(addr[0])

equipos.append(equipo)

#print(equipos)

#for f in equipos:

# print(f.ip)

x = data.decode().split("\n")

#print(len(x))

if x[0] == "ANNOUNCE":

for strlinea in x:

if strlinea != "ANNOUNCE":

#print(strlinea)

h = strlinea.split("\t")

if len(h) == 3 :

Archiv = Objetos.Archivo(0, "0","0","0")

for fl in equipo.archivos:

if fl.md5 == h[2]:

Archiv = fl

if Archiv.md5 == "0":

for fl in listaArchivos:

if fl.md5 == h[2]:

fl.ips.append(addr[0])

Archiv = fl

if Archiv.md5 == "0":

Objetos.Archivo.contidArchivo = Objetos.Archivo.contidArchivo + 1

Archiv = Objetos.Archivo(Objetos.Archivo.contidArchivo, h[0], h[1], h[2])

Archiv.ips.append(addr[0])

equipo.archivos.append(Archiv)

listaArchivos.append(Archiv)

else:

equipo.archivos.append(Archiv)

def anunciarArchivos():

sock = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM, socket.IPPROTO\_UDP)

sock.setsockopt(socket.SOL\_SOCKET, socket.SO\_BROADCAST, 1)

# Set a timeout so the socket does not block indefinitely when trying to receive data.

sock.settimeout(0.2)

while True:

mensaje = "ANNOUNCE\n"

archivos = ls("compartida")

stringarc = ""

for arc in archivos:

stringarc = ""

stringarc = stringarc + arc

sizefile = os.stat("compartida\\" + arc).st\_size

md5suma = hashlib.md5(open("compartida\\" + arc, 'rb').read()).hexdigest()

mensaje = mensaje + stringarc + "\t" + str(sizefile) + "\t" + md5suma + "\n"

sock.sendto( mensaje.encode() , ('<broadcast>', 2020))

#print("mensaje enviado al puerto 2020 (anunciarArchivos)\n")

aleatorio = random()

# print( "\n numero aleatorio de 0 a 1 " + str(aleatorio))

time.sleep(10 + aleatorio)

# Escucha conecciones de descarga y abre un nuevo hilo

# con destino el metodo utilesFiles.atenderCliente

def servidorTCP():

misocket = socket.socket()

misocket.bind((ipEsteEquipo, 2020))

misocket.listen(5)

print(" Nueva conexion estableida")

num\_hilo = 0

while True:

conexion, addr = misocket.accept()

print(" Nueva conexion estableida")

print(addr)

num\_hilo = num\_hilo + 1

hilo = threading.Thread(target=utilesFiles.atenderCliente,

args=(num\_hilo, conexion, addr))

hilo.start()

def descargarParteDeArchivo(parte , archivoToDescargar, desde , hasta , ip ):

print(" -parte : " + str(parte) + " " + archivoToDescargar.nombre + " - desde : " + str(desde) + " - hasta : " + str(hasta) + " - ip : " + str(ip) )

hilo = threading.Thread(target=utilesFiles.clienteTCP,

args=( archivoToDescargar, parte , desde , hasta, ip ),

kwargs={})

hilo.start()

return " -parte : " + str(parte) + " " + archivoToDescargar.nombre + " - desde : " + str(desde) + " - hasta : " + str(hasta) + " - ip : " + str(ip) + " \r\n"

def descargarArchivo(archivoToDescargar):

cantidadHost = len(archivoToDescargar.ips)

print(" - cantidad Host : " + str(cantidadHost) )

print(" - tamanio : " + str(archivoToDescargar.largo))

total = int( archivoToDescargar.largo)

tamBlock = 124

cant = int( total / tamBlock)

rest = total - ( cant \* tamBlock)

if rest > 0 :

cant = cant + 1

print(" - cantidad " + str(cant))

print(" - rest " + str(rest))

desde = 0

strret = ""

for i in range(cant):

ipnumero = i % cantidadHost

if i == cant - 1 :

hasta = desde + rest

else:

hasta = desde + tamBlock

strret = strret + descargarParteDeArchivo( i , archivoToDescargar, desde , hasta , archivoToDescargar.ips[ipnumero])

desde = desde + tamBlock

return strret

# Busca el archivo a descargar a partir del nuemero

def ObtenerArchivoADescargar(numeroArchivo):

archivoToDescargar = Objetos.Archivo(0, "0","0","0")

ipsToDownload = []

for arch in listaArchivos:

if arch.idArchivo == int(numeroArchivo):

archivoToDescargar = arch

print("Descargando archivo " + archivoToDescargar.nombre)

for eq in equipos:

for arc in eq.archivos:

print(eq.ip + " - " + arc.nombre + " - " + str(arc.idArchivo))

destinos = ""

cont = 0

for ip in archivoToDescargar.ips:

destinos = destinos + ip

cont = cont + 1

if len(archivoToDescargar.ips) > cont:

destinos = destinos + " , "

return "Descargando archivo : ----> '" + archivoToDescargar.nombre + "' \r\nDesde los host : " + destinos + "\r\n" , archivoToDescargar

# Solicita la descarga de un archivo

def clienteTCP():

misocket = socket.socket()

misocket.connect(('localhost', 8000))

fileMD5 = "ute.pdf"

start = 0

size = 1024

solicitud = "DOWNLOAD\n"

solicitud = solicitud + "<fileMD5>\n"

solicitud = solicitud + "<" + str(start) + ">\n"

solicitud = solicitud + "<" + str(size) + ">\n"

misocket.send(str.encode(solicitud))

respuesta = misocket.recv(1024)

print(respuesta.decode())

respuesta = misocket.recv(200000)

print(respuesta.decode())

print(len(respuesta))

misocket.close()

def ls(ruta='.'):

dir, subdirs, archivos = next(walk(ruta))

return archivos

def terminalConsola():

host = '' # (2)

port = 23 # (3)

s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) # (4)

s.bind((host, port)) # (5)

s.listen(1024) # (7)

conn, addr = s.accept() # (9)

print('Connected with ' + addr[0] + ':' + str(addr[1])) # (10)

data = ""

concat = ""

enviado = 0

fin = 0

barran = 0

while (fin == 0):

data = ""

data = conn.recv(1024).decode() # 11

if (data.find('\n') > -1):

barran = 1

else:

concat = concat + data

# if (concat.find('\n') != -1):

# print("Contains given substring ")

#else:

print(data + "--" + concat)

if enviado == 0:

enviado = 1

conn.sendall(data.encode()) # 1

if concat == 'list' and barran == 1:

barran = 0

concat = ""

strval = getListArchivos()

strMen = "\r\n" + strval + "\r\n"

conn.sendall(strMen.encode()) # 1

if concat == 'exit' and barran == 1:

break

if concat.find('get ') == 0 and barran == 1:

retorno , archivo = ObtenerArchivoADescargar(concat.replace('get ', ''))

retorno = retorno + descargarArchivo(archivo)

conn.sendall(retorno.encode())

barran = 0

concat = ""

conn.close()

s.close() # 14

print(data)

if concat == 'asds':

v = utilesFiles.cantidadPartes('ute.pdf')

cont = utilesFiles.ObtenerContenidoArchivo('ute.pdf')

print(v)

for i in range(v):

desde = (i) \* 1024

parte = cont[desde:(desde + 1024)]

print(desde)

print(parte)

print(len(parte))

utilesFiles.guardarParte(parte, 'ute.pdf', str(i))

if \_\_name\_\_ == '\_\_main\_\_':

Objetos.Archivo.contidArchivo = 0

ipEsteEquipo = [ip for ip in socket.gethostbyname\_ex(socket.gethostname())[2] if not ip.startswith("127.")][:1][0]

# Hilo para hacer los anuncios de archivos UDP

hiloAnuncios = threading.Thread(target=anunciarArchivos, args=())

hiloAnuncios.start()

# Hilo para escuchar los anuncios

hiloEscuchaAnuncios = threading.Thread(target=escucharAnuncios, args=())

hiloEscuchaAnuncios.start()

# Hilo para la terminal telnet

hiloterminalConsola = threading.Thread(target=terminalConsola, args=())

hiloterminalConsola.start()

# Hilo para responder solicitudes de descarga

hiloservidorTCP = threading.Thread(target=servidorTCP, args=())

hiloservidorTCP.start()