# Tarea 1
# Distribuidos

Profesor: Mauricio Solar
Integrantes:
Vicente Pacheco
Eduardo Padilla

# Problema 1

Descripción: Se debe realizar una conexión cliente-servidor, en donde el cliente le manda un mensaje al servidor y éste debe responder de vuelta.

Implementación: En primer lugar el servidor se levanta en el puerto 5000 para que luego el cliente se conecte a este a través de un socket. Establecida la conexión, el cliente manda un mensaje hacia el servidor con la ayuda de la función **sendall** de la librería socket para enviar mensajes, y este último lo recibe gracias a la función **recv** para recibir los mensajes, luego el servidor le envía al cliente un mensaje de vuelta como respuesta. Finalmente cuando terminan de enviarse mensajes, se cierran los sockets. Además, los mensajes intercambiados deben guardarse en un archivo txt en el servidor y cliente respectivamente.

# Cliente

```python
import socket

# Definir Host y Puerto
HOST = 'servidor'
PORT = 5000

# Crear archivo
f = open ('respuestas.txt','w')
f.write("Respuesta\n\n")

# Crear socket y conectar al Servidor
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((HOST, PORT))

# Numero de mensajes a enviar
NumMensajes = 10

# Contador de mensajes
Cont = 0

# Acciones a realizar con el Servidor
while(Cont < NumMensajes):
    # Se escribe un mensaje al Cliente
    texto = "Mensaje desde el Cliente " + (str(Cont+1))
    s.sendall(texto.encode())

    # Se recibe mensaje desde el Servidor
    pipe = s.recv(1024)
    msg = str(pipe, 'utf-8')

    # El mensaje se printea y se escribe en el archivo
    print("La respuesta del Servidor es: %s" %msg)
    f.write('{0}\n'.format(msg))

    Cont += 1

# Cerrar el archivo y el socket
f.close()
s.close()
```

# Servidor

```python
import socket

# Definir Host y Puerto
HOST = 'servidor'
PORT = 5000

# Crear archivo
f = open ('log.txt','w')
f.write("IP          Mensaje\n\n")

# Crear socket
#AF_INET para IPv4 y SOCK_STREAM para TCP
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((HOST, PORT))
s.listen()
conn, addr = s.accept()

# Obtener IP
nombre_equipo = socket.gethostname()
IP = socket.gethostbyname(nombre_equipo)

Cont = 0

# Conectarse con el Cliente
while (True):
    # Espera conexion del Cliente
    print("Esperando peticiones...")

    # Se recibe mensaje desde el Cliente
    pipe = conn.recv(1024)
    msg = str(pipe, 'utf-8')

    # Finalizar conexion
    if not pipe:
        break

    # El mensaje se printea y se escribe en el archivo junto con la IP
    print("El mensaje del Cliente es: %s" %msg)
    f.write('{0} {1}\n'.format(IP, msg))

    # Se responde al Cliente
    texto = "Respuesta " + str((Cont+1)) + " desde el Servidor"
    conn.sendall(texto.encode())
    Cont += 1
# Cerrar el archivo y el socket
f.close()
s.close()
```

# Problema 2

Descripción: Se debe enviar un mensaje desde el cliente con el fin de que sea guardado en un almacenamiento al azar, para esto se requiere de un servidor intermedio que re-dirija el mensaje hacia un almacenamiento, guardando la información de qué mensaje y donde se guardó.

Implementación: Para esto se crea una conexión entre el cliente y el servidor (headnode), el cual a su vez actúa como cliente, conectándose con 3 nodos de almacenamiento (datanodes). Luego, el cliente tiene la opción de enviar un mensaje, cerrar cualquiera de los 3 datanodes, o cerrar la conexión por completo. Finalmente, el headnode recibe como retorno la dirección en donde se guardó el mensaje, y le traspasa esta información al cliente.

# Cliente

```python
import socket
import time

# Definir Host y Puerto
#HOST = 'localhost'
HOST = 'headnode'
PORT = 5000

# Crear archivo
f = open ('registro_cliente.txt','w')
f.write("Num Datanode           Mensaje\n\n")

# Crear socket y conectar al Servidor
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((HOST, PORT))

flag = 0

# Acciones a realizar con el Servidor
while(True):
    # Menu de opciones
    print("Seleccione una opcion:")
    print("1. Enviar mensaje al Headnode")
    print("2. Finalizar conexion con el Datanode1")
    print("3. Finalizar conexion con el Datanode2")
    print("4. Finalizar conexion con el Datanode3")
    print("5. Finalizar conexion completa")
    # Se elige una opcion
    # Si la opcion es 1, se escribe un mensaje al Servidor y se espera una respuesta
    if(opcion == '1'):
        s.sendall(opcion.encode())
        texto = "Mensaje"
        time.sleep(1)
        s.sendall(texto.encode())

        # Se recibe mensaje desde el Servidor
        pipe = s.recv(1024)
        msg = str(pipe, 'utf-8')

        # El mensaje se escribe en el archivo
        f.write('{0}\n'.format(msg))

    elif(opcion == '2'):
        texto = "2"
        s.sendall(texto.encode())
        print("Datanode1 finalizado")
    elif(opcion == '3'):
        texto = "3"
        s.sendall(texto.encode())
        print("Datanode2 finalizado")
    elif(opcion == '4'):
        texto = "4"
        s.sendall(texto.encode())
        print("Datanode3 finalizado")
    # Si la opcion es 5, se finaliza la conexion con el Servidor
    elif(opcion == '5'):
        print("Conexion finalizada")
        break
    else:
        print("Opcion invalida, vuelva a intentar")

# Cerrar el archivo y el socket
f.close()
s.close()
```

# Headnode

```python
import socket
import random
import threading
import time
import sys

# Definir Host y Puerto
#HOST = 'localhost'
HOST = 'headnode'
PORT = 5000

PORT1 = 5001
PORT2 = 5002
PORT3 = 5003

ListaVivos = [0,0,0]
# Crear archivo
f = open ('registro_server.txt','a')
f.write("Num Datanode           Mensaje\n\n")
f2 = open ('heartbeat_server.txt','w')

# Crear socket cliente
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #AF_INET para IPv4 y SOCK_STREAM para TCP
s.bind((HOST, PORT))
s.listen()
conn, addr = s.accept()

# Crear socket datanode1
s1 = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
#s1.connect(("localhost", PORT1))
s1.connect(("datanode1", PORT1))
ListaVivos[0] = 1

# Crear socket datanode2
s2 = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
#s2.connect(("localhost", PORT2))
s2.connect(("datanode2", PORT2))
ListaVivos[1] = 1

# Crear socket datanode3
s3 = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
#s3.connect(("localhost", PORT3))
s3.connect(("datanode3", PORT3))
ListaVivos[2] = 1


def Alive(Puerto1, Puerto2, Puerto3, Socket1, Socket2, Socket3, Archivo):
    while(True):
        global flag
        global ListaVivos

        confirmo = "confirmar"

        if(ListaVivos[0] == 1):
            Socket1.sendall(confirmo.encode())
        if(ListaVivos[1] == 1):
            Socket2.sendall(confirmo.encode())
        if(ListaVivos[2] == 1):
            Socket3.sendall(confirmo.encode())

        time.sleep(5)

        try:
            respuesta1 = Socket1.recv(1024)
            ACK1 = str(respuesta1, 'utf-8')
            Archivo.write("Datanode1 vivo\n")
            if(ACK1 == ""):
                ListaVivos[Puerto1 - 5001] = 0
                Archivo.write("Datanode1 no vivo\n")
        except:
            print("")
```

# Headnode (continuación)

```
        try:
            respuesta2 = Socket2.recv(1024)
            ACK2 = str(respuesta2, 'utf-8')
            Archivo.write("Datanode2 vivo\n")
            if(ACK2 == ""):
                ListaVivos[Puerto2 - 5001] = 0
                Archivo.write("Datanode2 no vivo\n")
        except:
            print("")

        try:
            respuesta3 = Socket3.recv(1024)
            ACK3 = str(respuesta3, 'utf-8')
            Archivo.write("Datanode3 vivo\n")
            if(ACK3 == ""):
                ListaVivos[Puerto3 - 5001] = 0
                Archivo.write("Datanode3 no vivo\n")
        except:
            print("")
        #result1 = Socket1.connect_ex(('localhost', Puerto1))
        #result2 = Socket2.connect_ex(('localhost', Puerto2))
        #result3 = Socket3.connect_ex(('localhost', Puerto3))

        if(flag):
            break

    # Obtener IP
    nombre_equipo = socket.gethostname()
    IP = socket.gethostbyname(nombre_equipo)

    # Espera conexion del Cliente
    print("Esperando peticiones...")
```

```
# Conectarse con el Cliente
while True:
    flag = False
    t = threading.Thread(target = Alive, args = (PORT1, PORT2, PORT3, s1, s2, s3, f2))
    t.start()
    # Se recibe mensaje desde el Cliente
    pipe = conn.recv(1024)
    msg = str(pipe, 'utf-8')
    if(msg == '1'):
        pipeCliente = conn.recv(1024)
        msgCliente = str(pipeCliente, 'utf-8')
        while(True):
            NumDatanode = random.choice([1,2,3])
            if(ListaVivos[NumDatanode-1] == 1):
                break
        if(NumDatanode == 1 and ListaVivos[0]):
            s1.sendall(msgCliente.encode())
            pipe1 = s1.recv(1024)
            ACK = str(pipe1, 'utf-8')
        elif(NumDatanode == 2 and ListaVivos[1]):
            s2.sendall(msgCliente.encode())
            pipe2 = s2.recv(1024)
            ACK = str(pipe2, 'utf-8')
        elif(NumDatanode == 3 and ListaVivos[2]):
            s3.sendall(msgCliente.encode())
            pipe3 = s3.recv(1024)
            ACK = str(pipe3, 'utf-8')
        if(ACK != "confirmo"):
            print("Mensaje guardado correctamente")
        MensajeCliente = str(NumDatanode) + '                    ' + msgCliente
        f.write('{0} \n'.format(MensajeCliente))
        conn.sendall(MensajeCliente.encode())
```

# Headnode (continuación)

```
139     elif(msg == '2'):
140         matar = "kill"
141         s1.sendall(matar.encode())
142         pipe1 = s1.recv(1024)
143     elif(msg == '3'):
144         matar = "kill"
145         s2.sendall(matar.encode())
146         pipe2 = s2.recv(1024)
147     elif(msg == '4'):
148         matar = "kill"
149         s3.sendall(matar.encode())
150         pipe3 = s3.recv(1024)
151     # Finalizar conexion
152     elif not pipe:
153         flag = True
154         break
155 
156 # Cerrar el archivo y el socket
157 f.close()
158 s.close()
159 s1.close()
160 s2.close()
161 s3.close()
162 sys.exit()
```

# Datanode

```python
import socket

# Definir Host y Puerto
#HOST = 'localhost'
HOST = 'datanode1'
PORT = 5001

# Crear archivo
f = open ('data.txt','w')
f.write("Mensaje\n\n")

# Crear socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #AF_INET para IPv4 y SOCK_STREAM para TCP
s.bind((HOST, int(PORT)))
s.listen()
conn, addr = s.accept()

# Obtener IP
nombre_equipo = socket.gethostname()
IP = socket.gethostbyname(nombre_equipo)

# Espera conexion del Headnode
print("Esperando peticiones...")

# Conectarse con el Headnode
while True:

    # Se recibe mensaje desde el Headnode
    pipe = conn.recv(1024)
    msg = str(pipe, 'utf-8')
    if(msg == "confirmar"):
        resp = "confirmo"
        conn.sendall(resp.encode())
    elif(msg == "kill"):
        break
    else:
        # Finalizar conexion
        if not pipe:
            break

        # El mensaje se escribe en el archivo
        f.write('{0} \n'.format(msg))
        respuesta = "Mensaje guardado correctamente"
        conn.sendall(respuesta.encode())

    # Se responde al Cliente
    #print("Escriba la respuesta para el Cliente:")
    #texto = input()
    #conn.sendall(texto.encode())

# Cerrar el archivo y el socket
f.close()
s.close()
```