

# Informe del Trabajo Práctico N.º 4

## Parte B

## Transferencia de Archivos mediante Sockets TCP

### Introducción

Este trabajo práctico tuvo como objetivo implementar una aplicación basada en **sockets TCP** para realizar la transferencia de archivos entre un servidor y un cliente. La actividad permitió reforzar los conceptos fundamentales de la **capa de transporte**, como el establecimiento de conexiones confiables, el envío y recepción de datos, y el manejo de archivos desde una aplicación de red.

### Actividad 1: Transferencia de archivos sobre sockets TCP Desarrollo

Se desarrollaron dos scripts en Python 3:

- **tp4B\_Server.py**: ejecutado por el servidor, espera conexiones de clientes y les envía un archivo seleccionado.
- **tp4B\_Cliente.py**: se conecta al servidor, recibe el archivo y lo guarda localmente.

### Funcionamiento general:

1. El **servidor** espera conexiones en una IP y puerto determinados.
2. Al establecerse la conexión, solicita al usuario del servidor que indique **la ruta del archivo** a transferir.
3. El servidor **envía el contenido del archivo en bloques** (chunks) de datos al cliente usando TCP.
4. El **cliente recibe los datos y los guarda** en un archivo local.
5. Una vez finalizada la transferencia, ambas partes cierran la conexión.

- **Al realizar la conexión entre el servidor y el cliente**

Server:

Integrantes: Martina Nahman y Emiliano Germani

```
8  PORT = 60000
9
10 def servidor():
11     sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
12     sock.bind((HOST, PORT))

PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  PUERTOS  Python: tp4B_Server

PS C:\Users\Usuario\Documents\MARTI\FING\2025\Redes\TPS\tp4\tp4b> & C:/Users/Usuario/AppData/Local/
pps/python3.11.exe c:/Users/Usuario/Documents/MARTI/FING/2025/Redes/TPS/tp4/tp4b/tp4B_Server.py
[+] Servidor escuchando en 0.0.0.0:60000
[+] Conexión establecida desde ('192.168.1.69', 57674)
Ingrese la ruta del archivo a enviar: "C:\Users\Usuario\Desktop\CV Nahman Martina.pdf"
```

Cliente:

```
19
20     nombre_archivo, tamaño_str = encabezado.split("|")
21     tamaño_total = int(tamaño_str)
22
23     # Confirmar que estamos listos

PROBLEMAS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\Piruc\OneDrive\Escritorio\tp4b> & C:/Users/Piruc/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/Piruc/
ente.py
Ingrese la IP del servidor: 192.168.1.18
[+] Conectado al servidor.
```

- Se probó la aplicación en una red local. Si el servidor intenta enviar un archivo que no existe

Server:

```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  PUERTOS  Python: tp4B

PS C:\Users\Usuario\Documents\MARTI\FING\2025\Redes\TPS\tp4\tp4b> & C:/Users/Usuario/AppData/
pps/python3.11.exe c:/Users/Usuario/Documents/MARTI/FING/2025/Redes/TPS/tp4/tp4b/tp4B_Server
[+] Servidor escuchando en 0.0.0.0:60000
[+] Conexión establecida desde ('192.168.1.69', 57674)
Ingrese la ruta del archivo a enviar: "C:\Users\Usuario\Desktop\CV Nahman Martina.pdf"
[!] El archivo no existe. Cerrando conexión.
PS C:\Users\Usuario\Documents\MARTI\FING\2025\Redes\TPS\tp4\tp4b> 
```

Integrantes: Martina Nahman y Emiliano Germani

Cliente:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\Piruc\OneDrive\Escritorio\tp4b> & C:/Users/Piruc/AppData/Local/M
ente.py
Ingrese la IP del servidor: 192.168.1.18
[+] Conectado al servidor.
[!] Error del servidor: ERROR: Archivo no encontrado.
PS C:\Users\Piruc\OneDrive\Escritorio\tp4b> 
```

- Si el servidor ejecutó correctamente el envío de archivos de texto

Server:

```
tp4B_Server.py X  tp4B_Cliente.py

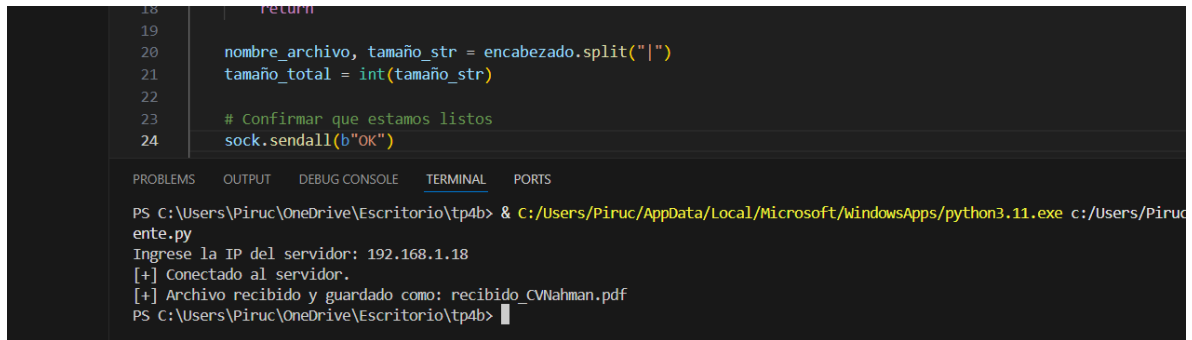
tp4b > tp4B_Server.py > servidor
10  def servidor():
15
16      conn, addr = sock.accept()
17      print(f"[+] Conexión establecida desde {addr}")
18
19      # Pedir al usuario del servidor el nombre del archivo
20      nombre_archivo = input("Ingrese la ruta del archivo a enviar: ")
21
22      # Verificar si el archivo existe
23      if not os.path.exists(nombre_archivo):
24          print("[!] El archivo no existe. Cerrando conexión.")
25      sock.close()

PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  PUERTOS

PS C:\Users\Usuario\Documents\MARTI\FING\2025\Redes\TPS\tp4\tp4b> & C:/Users/U
pps/python3.11.exe c:/Users/Usuario/Documents/MARTI/FING/2025/Redes/TPS/tp4/tp
[+] Servidor escuchando en 0.0.0.0:60000
[+] Conexión establecida desde ('192.168.1.69', 57818)
Ingrese la ruta del archivo a enviar: CVNahman.pdf
[+] Archivo 'CVNahman.pdf' enviado con éxito.
PS C:\Users\Usuario\Documents\MARTI\FING\2025\Redes\TPS\tp4\tp4b> 
```

Integrantes: Martina Nahman y Emiliano Germani

Cliente:



```
18         return
19
20     nombre_archivo, tamaño_str = encabezado.split("|")
21     tamaño_total = int(tamaño_str)
22
23     # Confirmar que estamos listos
24     sock.sendall(b"OK")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\Piruc\OneDrive\Escritorio\tp4b> & C:/Users/Piruc/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/Piruc/ente.py
Ingrese la IP del servidor: 192.168.1.18
[+] Conectado al servidor.
[+] Archivo recibido y guardado como: recibido_CVNahman.pdf
PS C:\Users\Piruc\OneDrive\Escritorio\tp4b>
```

## Pruebas realizadas

Se probó la aplicación en una red local. El servidor ejecutó correctamente el envío de archivos de texto y binarios (PDF). El cliente recibió los archivos completos y sin corrupción.

## Conclusión

Este trabajo permitió consolidar conocimientos sobre programación de sockets TCP, flujo de datos binarios, y comunicación entre procesos en red. Además, se abordó el uso de funciones para manipulación de archivos, y se comprendió la importancia de enviar datos en bloques controlados para asegurar la integridad de la transmisión. Este tipo de aplicaciones constituye la base de protocolos reales como FTP.