FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Trabajo Práctico 1

[TA048] Redes
Primer cuatrimestre de 2025

Alumno	Padron	Email
AVALOS, Victoria	108434	vavalos@fi.uba.ar
CASTRO MARTINEZ, Jose Ignacio	106957	jcastrom@fi.uba.ar
CIPRIANO, Victor	106593	vcipriano@fi.uba.ar
DEALBERA, Pablo Andres	106858	pdealbera@fi.uba.ar
DIEM, Walter Gabriel	105618	wdiem@fi.uba.ar

Contents

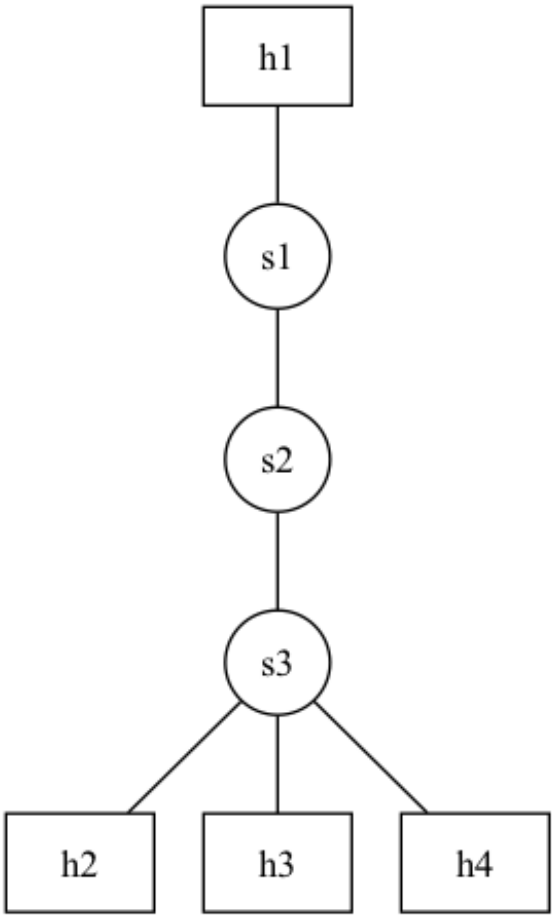
1 Introduccion

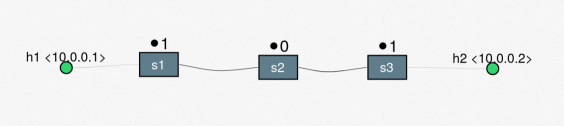
- La carga/descarga no va a conservar la metadata del archivo. Es decir, si yo descargo un archivo, ese archivo va a tener metadata como si yo hubiera creado el archivo desde cero usando 'touch archivo'.
- Si el cliente utiliza otro protocolo para comunicarse con el server, el server debe rechazar este pedido. ($\text{PROTOCOL}_{\text{MISSMATCH}}$). El header tendra un campo dedicado a esto.
- El argumento de FILENAME sera opcional, en caso de no estar, se utiliza el nombre original del archivo.
- Por simplicidad, vamos a guardar todos los archivos en DIRPATH sin ningun nivel de subdirectorios.
- Por simplicidad, vamos a tener un tamaño maximo de 2GB para la subida y descarga de archivos.
- Los archivos en proceso de escritura se van a escribir en una ubicacion temporal para evitar que se corrompan en la ubicacion que el cliente pidio.
- Usar seek y bufferear para leer el archivo. Leer con slices del buffer.

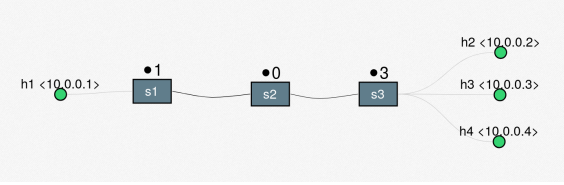
Caso borde: Dos clientes cargando y descargando el mismo archivo al mismo tiempo.

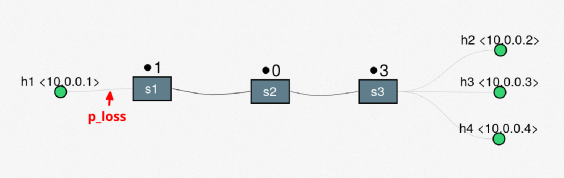
2 Implementacion

2.1 Topología









2.2 Especificación del protocolo Stop-and-Wait

2.2.1 General

Tamaño máximo de payload de: 61440 Bytes (60kB). Este número se obtiene teniendo en cuenta que el header de UDP posee un tamaño máximo de payload de 65535 (64kB), dejando espacio para los headers.

Se usa la nomenclatura S para mencionar al servidor y C para el cliente.

2.2.2 Handshake

La idea es usar este handshake para inicialización de recursos del servidor y check de protocolo.

Mensajes para caso Download y caso Upload:

1. C → S: con flag SYN para declarar una solicitud de conexion y el protocolo

Flujo normal (mismo protocolo):

2. S → C: con flag de SYN y ACK para declarar que se acepta la conexión y el puerto donde se va a escuchar el resto.
3. C → S: con flag ACK al mismo welcoming socket. Se declara la operación (OP) en el payload, que puede ser download (1) o upload (2).
4. S → C: ACK de la operación desde el connection socket

Flujo de error (distinto protocolo):

2. S → C: con flag FIN para denegar la conexión por usar un protocolo distinto.

Se hace una transferencia de puerto para que el welcoming socket se encargue solamente de establecer conexiones y el nuevo puerto maneje la transferencia de datos del archivo. El último ACK de parte del cliente asegura que se recibió el puerto donde se tiene que comunicar y es seguro hacer el cambio de socket.

2.2.3 Etapa de configuración y Transferencia

El cliente ya sabe que tiene que comunicarse con el nuevo puerto y el servidor ya sabe qué operación se quiere realizar.

_Mensajes para caso Download:

3. Mensaje 3 C → S: filename

Flujo Normal:

4. S → C: ACK + comienzo de datos (piggybacked)
5. C → S: ACK
6. S → C: continuacion de datos
7. C → S: ACK

...

Flujo de error (no existe un archivo con ese nombre):

1. S → C: FIN
2. C → S: ACK

3. C → S: FIN
4. S → C: ACK

_Mensajes para caso Upload:

3. C → S: filename

Flujo de error (ya existe un archivo con ese nombre):

1. S → C: FIN
2. C → S: ACK
3. C → S: FIN
4. S → C: ACK

Flujo normal:

4. S → C: ACK
5. C → S: filesize

Flujo de error (No hay más espacio en disco):

1. S → C: FIN
2. C → S: ACK
3. C → S: FIN
4. S → C: ACK

Flujo normal:

6. S → C: ACK
7. C → S: comienzo de datos
8. S → C: ACK
9. C → S: continuacion de datos

...

2.2.4 Cierre

El flag FIN va piggybacked con la última data para que sea más eficiente. El receptor confirma con un ACK + FIN para que el emisor sepa que le llego la informacion, y por si este se pierde está el último ACK para confirmar el cierre de parte del emisor.

Mensajes para caso Download:

1. S → C: ultima data, va piggybacked el flag FIN
2. C → S: ACK
3. C → S: FIN
4. S → C: ACK

Mensajes para caso Upload:

1. C → S: ultima data, va piggybacked el flag FIN
2. S → C: ACK
3. S → C: FIN
4. C → S: ACK