Tecnología Digital IV: Redes de Computadoras

Clase 6: Nivel de Aplicación - Parte 4

Lucio Santi & Emmanuel Iarussi

Licenciatura en Tecnología Digital
Universidad Torcuato Di Tella

27 de Marzo 2025

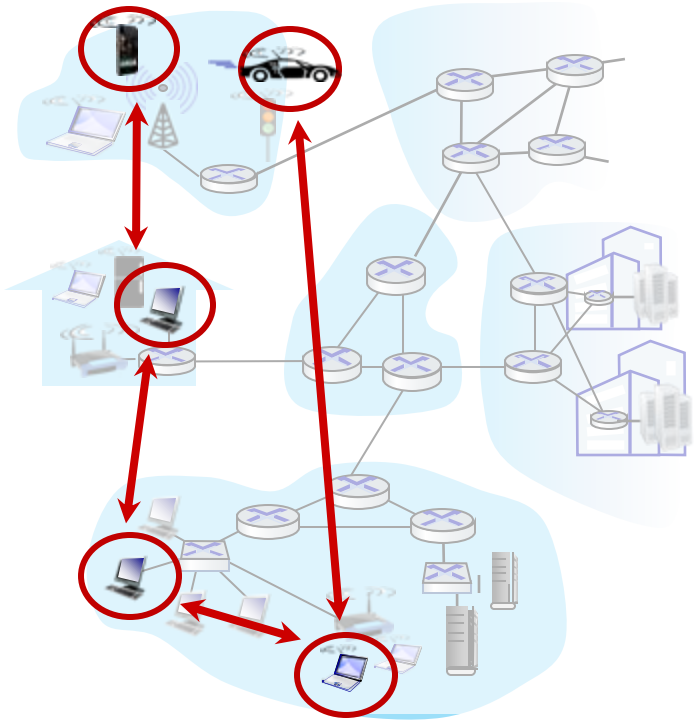
Agenda

- Protocolos de la capa de aplicación
  - Streaming de video: DASH
  - Distribución de archivos peer-to-peer (BitTorrent)
- Redes de distribución de contenido (CDNs)

Aplicaciones P2P - BitTorrent

Arquitectura *peer-to-peer* (P2P)

- No hay *un* servidor
- Comunicación directa entre hosts/*peers*
- Los peers solicitan/ofrecen servicios a otros peers
  - Auto-escalabilidad
- Los peers tienen conectividad intermitente; cambian direcciones IP
- Ejemplo: intercambio de archivos P2P (e.g. BitTorrent)

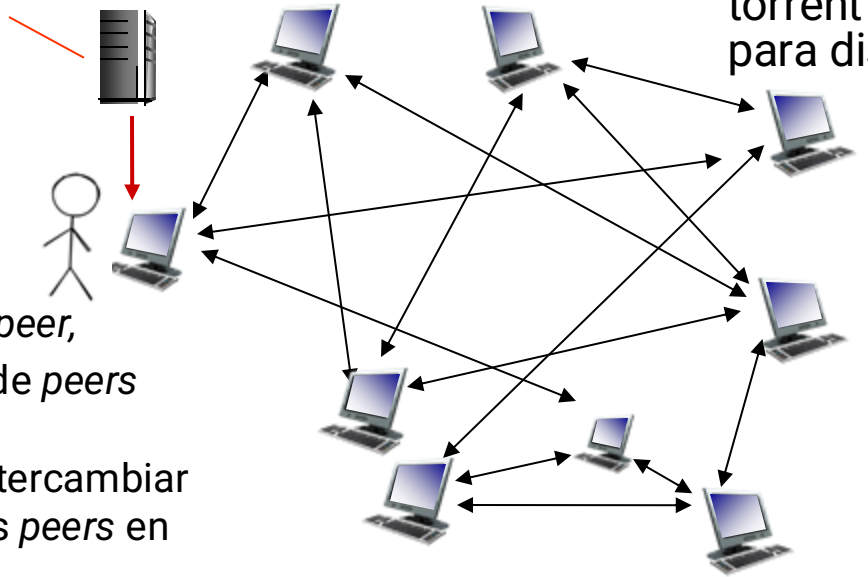


BitTorrent: distribución de archivos P2P

- Los archivos se dividen en *pieces* (o *chunks*) de 256 kB
- Cada *peer* de un **swarm** envía y recibe *chunks* de archivos

tracker: registra *peers* participando en un *swarm*

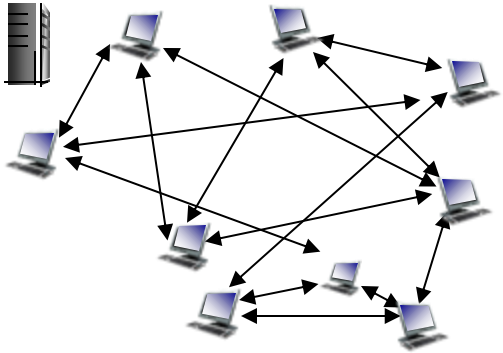
swarm: conjunto de *peers* intercambiando *chunks* de un torrent (descripción de archivos para distribuir)



Llega un nuevo *peer,*
obtiene la lista de *peers* del *tracker,*
y comienza a intercambiar *pieces* con otros *peers* en el *swarm*

BitTorrent: distribución de archivos P2P

- Cuando un *peer* se une a un *swarm*:
  - No posee *chunks* (los irá acumulando a lo largo del tiempo por medio de los demás *peers*)
  - Se registra en el *tracker* para obtener la lista de *peers* y se conecta a *peers* “vecinos”

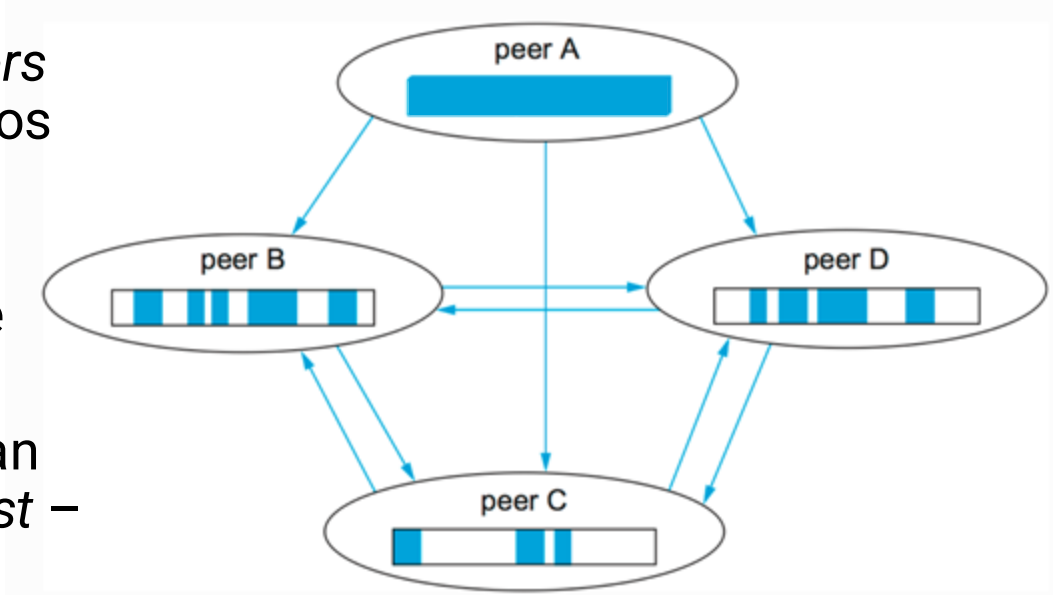


- Durante las descargas, los *peers* suben *chunks* a otros *peers*
- Los *peers* pueden cambiar el conjunto de *peers* con quienes se intercambian *chunks*
- *churn:* los *peers* pueden venir e irse dinámicamente
- Cuando un *peer* obtiene el archivo completo, puede irse (peer “egoísta”) o permanecer en el torrent (*peer* “altruista”)

BitTorrent: pedido y envío de *chunks*

Pedido de *chunks*

- En todo instante, distintos *peers* poseen diferentes subconjuntos de los *chunks* del archivo
- Periódicamente, cada *peer* solicita a los demás la lista de *chunks* que ellos poseen
- A partir de esta info se solicitan los *chunks* faltantes (*rarest first* – los más “inusuales” primero)
- A veces, la política es *random first*

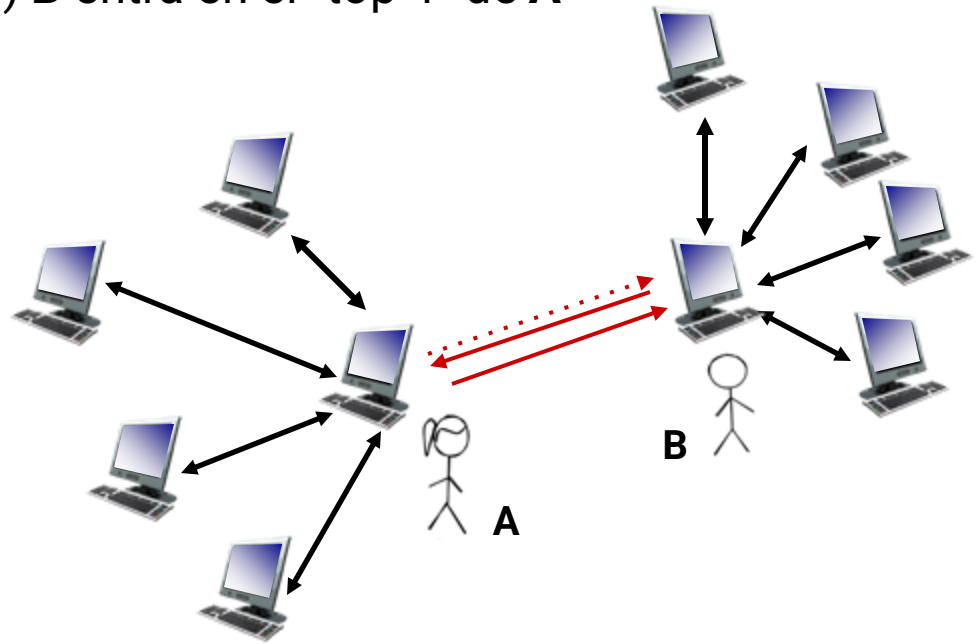


BitTorrent: *tit-for-tat*

Envío de *chunks*: *tit-for-tat*

- Un *peer* envía *chunks* a los cuatro *peers* que le envían *chunks* a mayor velocidad
  - Los demás *peers* no reciben nada (*choked peers*)
  - El "top 4" se reevalúa cada diez segundos
- Cada 30 segundos se selecciona aleatoriamente otro *peer* y se le envía *chunks*
  - "*optimistic unchoking*"
  - El *peer* seleccionado puede subirse al "podio" de los top 4

(1) **A** selecciona a **B** vía *optimistic unchoking*
(2) **A** entra en el "top 4" de **B**; **B** envía *chunks* a **A**
(3) **B** entra en el "top 4" de **A**



Streaming de video y CDNs

Streaming y CDNs: contexto

- El tráfico de streaming es el **mayor consumidor de ancho de banda en Internet**
  - Netflix, YouTube, Amazon Prime: 80% del tráfico residencial de ISPs (2020)
- Desafío: escala – cómo alcanzar a miles de millones de usuarios
- Desafío: heterogeneidad
  - Cada usuario tiene distintas características (e.g., cableado vs. móvil)
- Solución: infraestructura distribuida (a nivel de aplicación)

Video

- Secuencia de imágenes mostradas a una tasa constante
  - e.g., 24/30 imágenes por segundo
- Una **imagen** (digital) es un arreglo de píxeles (cada píxel se representa con bits)
- **Codificación**: utilizar redundancia dentro de y entre imágenes para reducir la cantidad de bits del video
  - espacial: dentro de la imagen
  - temporal: de una imagen a la siguiente

codificación espacial:
en vez de enviar *N* valores del mismo color, se envían sólo dos valores: el color (violeta) y la cantidad de repeticiones (*N*)

cuadro i

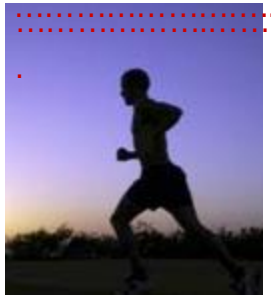
cuadro i+1

codificación temporal:
en vez de enviar el cuadro *i+1* completo, se envían sólo las diferencias respecto del cuadro *i*

Compresión de video

- CBR (*constant bit rate*): la tasa de codificación está fija
- VBR (*variable bit rate*): la tasa cambia de acuerdo a cuánto cambia la codificación espacial/temporal
- Ejemplos:
  - MPEG1 (CD-ROM) 1.5 Mbps
  - MPEG2 (DVD) 3-6 Mbps
  - MPEG4 (usado en Internet; 64Kbps – 12 Mbps)

codificación espacial:
en vez de enviar *N* valores del mismo color, se envían sólo dos valores: el color (violeta) y la cantidad de repeticiones (*N*)

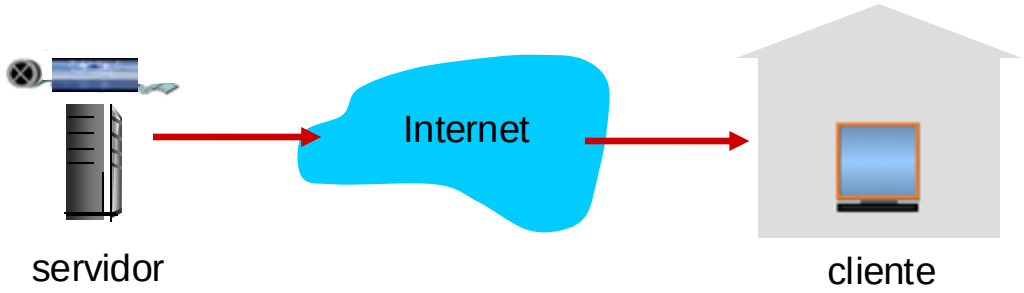


cuadro i

cuadro i+1

codificación temporal:
en vez de enviar el cuadro *i+1* completo, se envían sólo las diferencias respecto del cuadro *i*

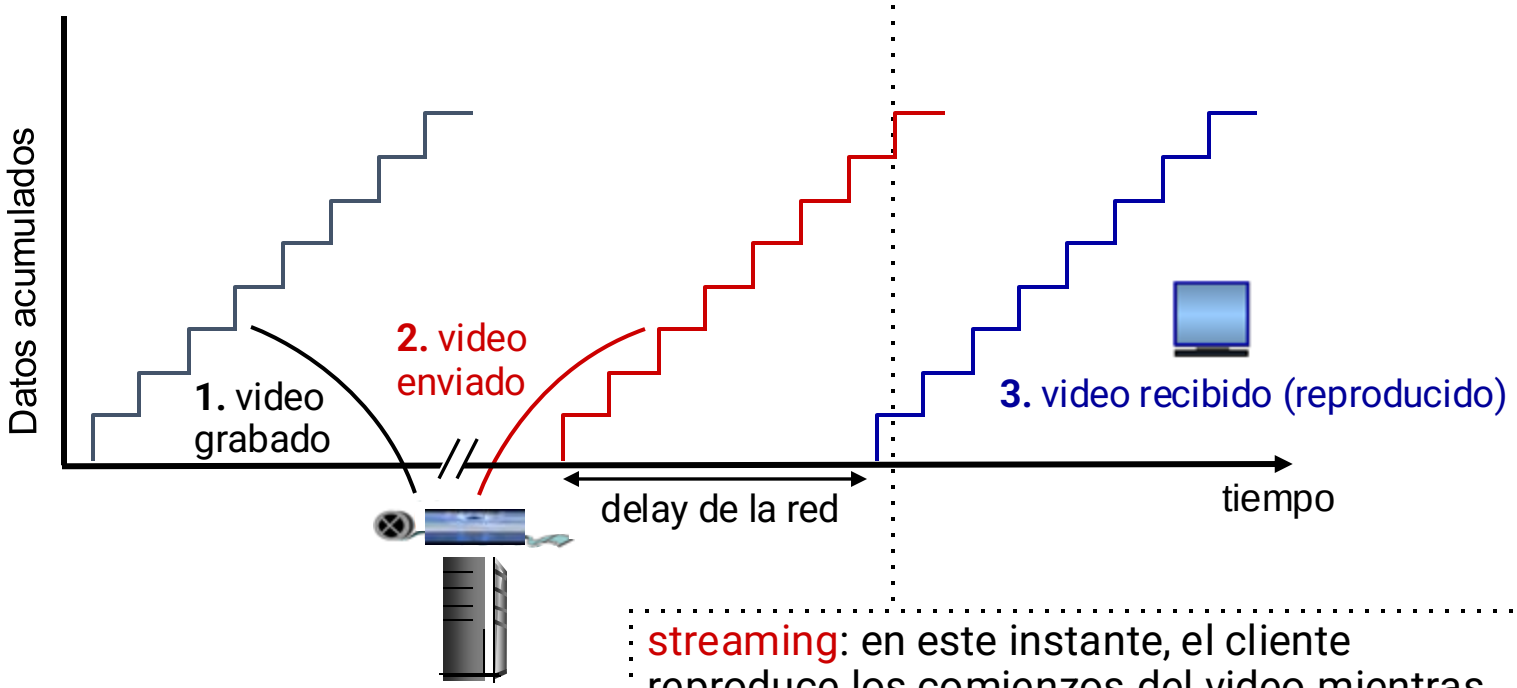
Streaming de video almacenado



Desafíos principales:

- El ancho de banda del servidor al cliente puede cambiar a lo largo del tiempo (con niveles de congestión cambiantes)
- La pérdida de paquetes y el delay por congestión pueden demorar la reproducción del video o deteriorar su calidad

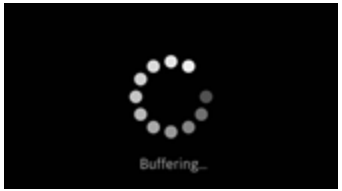
Streaming de video almacenado



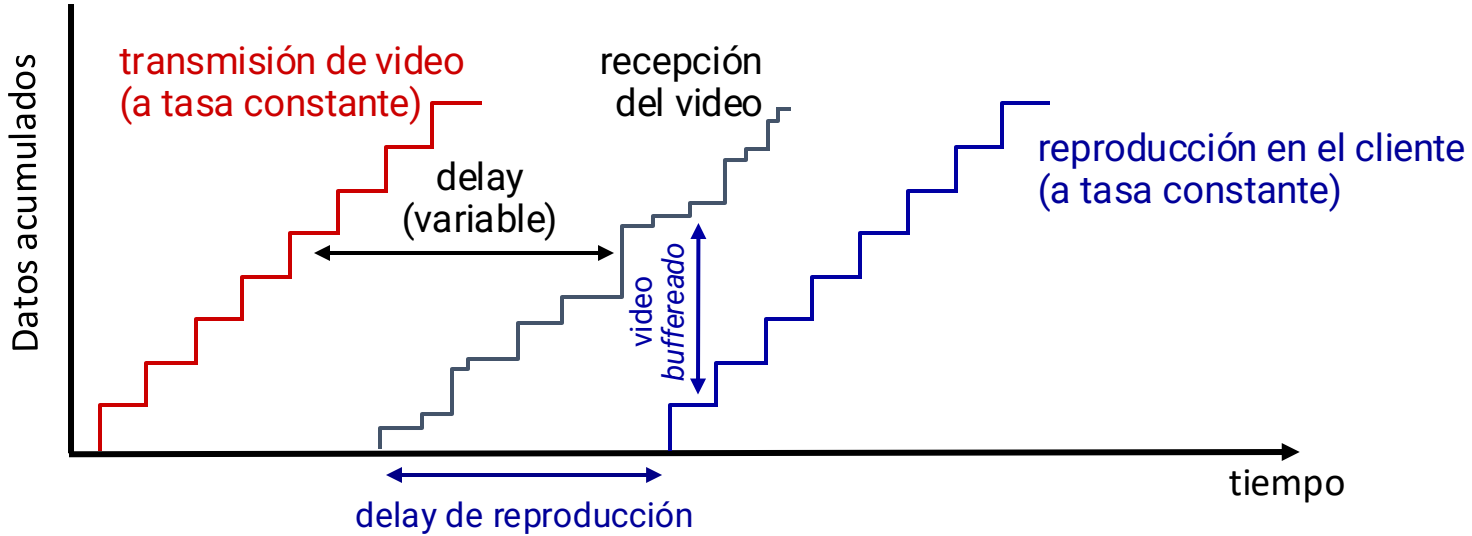
streaming: en este instante, el cliente reproduce los comienzos del video mientras que el servidor aún sigue enviando las porciones finales

Streaming: desafíos

- Reproducción continua: el tiempo de reproducción debe coincidir con el tiempo del video original
  - Los delays de la red son variables (*jitter*), de modo que será necesario hacer **buffering** en el cliente
- Interactividad: el cliente puede pausar, adelantar, rebobinar, etc.
- Pérdida de paquetes / retransmisiones



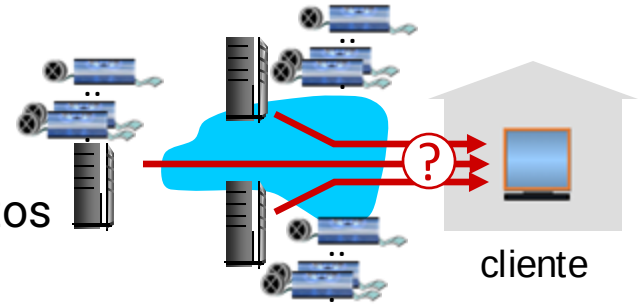
Streaming de video: *buffering*



- El **buffering** y el **delay de reproducción** “compensan” el delay de la red

Transmisión adaptable: DASH

Protocolo **DASH**: *Dynamic Adaptive Streaming over HTTP*



Servidor

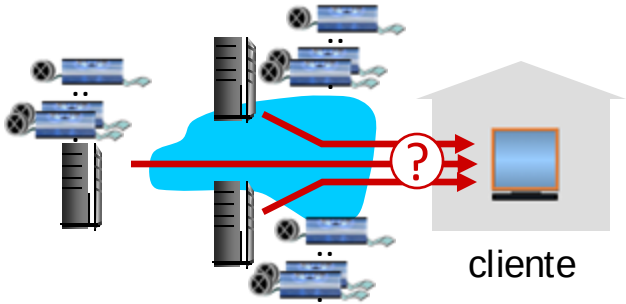
- Divide el archivo de video en *chunks*
- Cada *chunk* se codifica a distintas tasas
- Estas codificaciones se almacenan en archivos distintos
- Los archivos se replican en diversas CDNs
- Un *manifiesto* provee URLs para los distintos *chunks*

Cliente

- Estima periódicamente el ancho de banda disponible desde el servidor
- A través del manifiesto, solicita un *chunk* por vez
  - Elige la máxima tasa de codificación sostenible dado el ancho de banda actual
  - Puede escoger diferentes tasas en diferentes instantes de tiempo (y también desde distintos servidores)

Transmisión adaptable: DASH

- “Inteligencia” en el cliente: determina
  - Cuándo pedir un chunk (para evitar buffers vacíos o colapsados)
  - Cuál tasa de codificación pedir (mayor calidad cuando haya disponibilidad de mayor ancho de banda)
  - Dónde pedir un chunk (puede utilizar la URL de un servidor cercano o con un ancho de banda mayor)



Streaming de video = codificación + DASH + *buffering*

Redes de Distribución de Contenido (CDNs)

¿Cómo ofrecer streaming de **millones** de videos a cientos de miles de usuarios **simultáneos**?

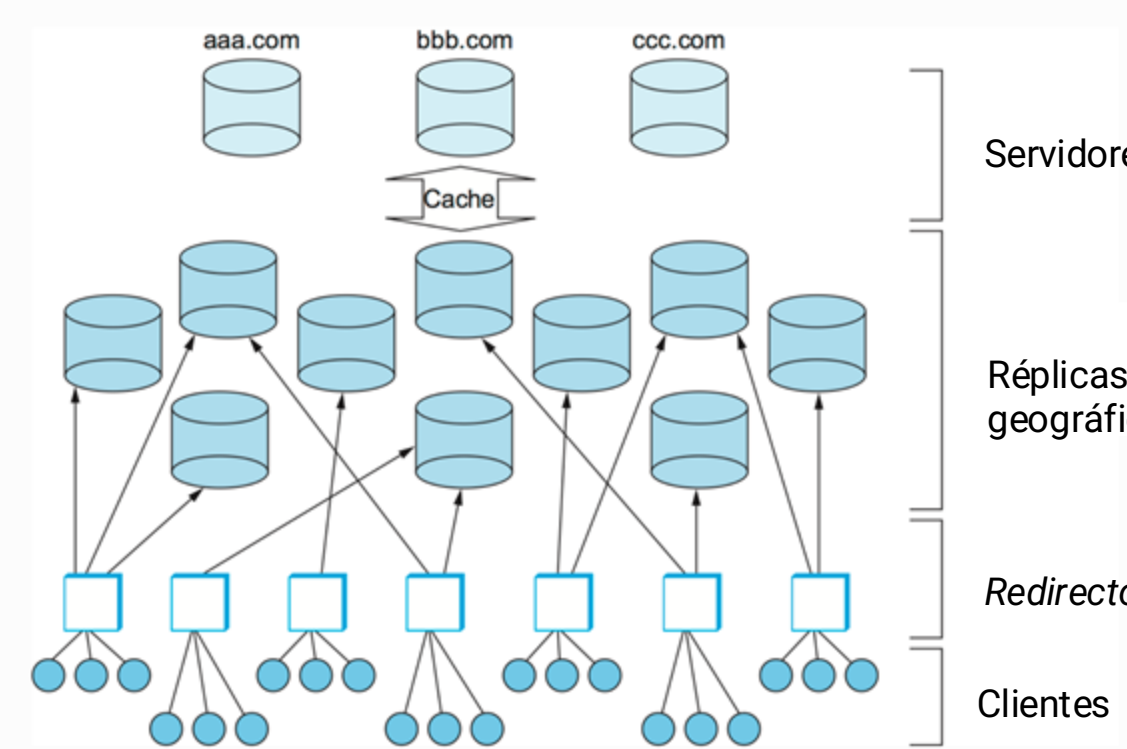
- Servidor “enorme” (?)
  - Punto único de falla
  - Congestión
  - Caminos largos (posiblemente congestionados) a muchos clientes distantes
  - No escala

Redes de Distribución de Contenido (CDNs)

¿Cómo ofrecer streaming de **millones** de videos a cientos de miles de usuarios **simultáneos**?

- Alternativa: almacenar múltiples copias de los videos en diversos sitios geográficamente distribuidos – CDNs
  - Incluso, con servidores ubicados en varias redes de acceso (ISPs)
    - Proximidad a los usuarios
    - Akamai: 360.000 servidores instalados en más de 135 países y en más de 1350 redes a lo largo del mundo

Componentes de una CDN



https://book.systemsapproach.org/applications/overlays.html#content-distribution-networks

Distribución de contenido multimedia

- Las CDNs almacenan copias del contenido multimedia (ej. *Breaking Bad*) en sus nodos
- Los usuarios, al solicitar el contenido, reciben un *manifiesto* del proveedor (ej. Netflix)
  - Mediante éste, el cliente puede obtener el contenido a la mayor tasa de transferencia soportada
  - Puede elegir una tasa o una copia distinta si ej. hay congestión