Nivel de aplicación

Email

Tienes tres componentes principales:

- User Agents
- Servidores de mail
- Simple Mail Transfer Protocol (SMTP)

User Agents

Permite redactar, editar, responder, reenviar y leer mensajes. Los mensajes son almacenados en un servidor. Algunos ejemplos son Outlook, Gmail, Yahoo, et cetera.

Servidores de mail

En los servidores tenemos:

- El buzón que guarda los mensajes entrantes para el usuario.
- Una cola de salida de los mensajes a ser enviados

SMTP

Permite enviar mensajes entre servidores. Es el principal protocolo del nivel de aplicación para los e-mails de Internet.

Cliente: servidor de mail emisor.

Servidor: servidor de mail recipiente.

Utiliza TCP para transferencias confiables entre el servidor de mail del emisor y el servidor de mail del recipiente (el servidor escucha en el puerto 25).

Hay tres fases:

- Handshaking. En ésta fase, el cliente SMTP indica la dirección e-mail del emisor (o sea, la persona que genera el mail) y la dirección del recipiente.
- Transferencia de mensajes. Como la conexión ya fue establecida, comienza la transferencia del mensaje por parte del emisor. SMTP puede confiar en que el servicio de transferencia de data es confiable y que el otro servidor de mail recibirá el mensaje sin ningún error. Si el emisor desea enviar más mensajes puede hacerlo a través de la misma conexión TCP ya establecida.
- Cierre. Si no hay más mensajes que enviar, el servidor le instruccciona a TCP para que cierre la conexión.

Interacciones de tipo comando/respuesta (Como HTTP) en formato ASCII. Las imágenes o videos binarios necesitan ser codificados a ASCII antes de ser enviados mediante SMTP. Así como es codificada, también necesita ser decodificada luego de su transporte.

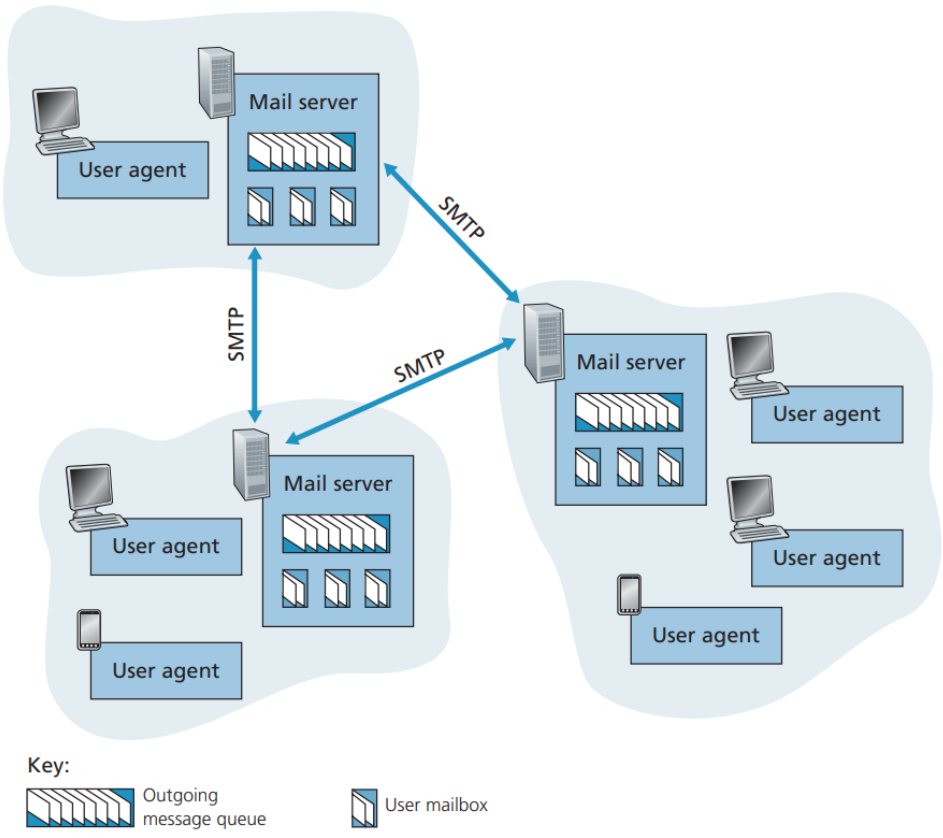


Figure 2.14 ♦ A high-level view of the Internet e-mail system

Escenario: envío y lectura de mails

1) Alice utiliza su *User Agent* (UA) para redactar un mail a `bob@utdt.edu`

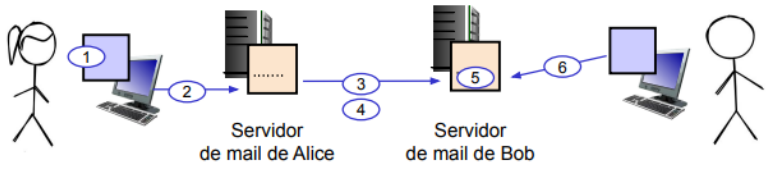
2) El UA envía el mensaje a su servidor vía SMTP; el mensaje queda encolado hasta ser transmitido

3) El cliente SMTP abre una conexión TCP con el servidor de mail de Bob

4) El cliente SMTP envía el mensaje de Alice por la conexión TCP

5) El servidor de mail del usuario B coloca el mensaje recibido en el buzón de Bob

6) Bob utiliza su UA para leer el mensaje



TD4 2023 7

Si al tratar de enviar un mensaje, el servidor del recipiente está caído, el mensaje permanecerá en el servidor del emisor y esperará a intentar de nuevo establecer una conexión.

Formato de los mensajes

Líneas de headers:

To:

From:

Subject

Body: el mensaje en sí mismo; sólo caracteres ASCII permitidos.

SMTP vs. HTTP

SMTP	HTTP
Push del cliente	Pull del cliente
Ambos tienen interacciones	Ambos tienen interacciones

comando/respuesta en ASCII y códigos de status	comando/respuesta en ASCII y códigos de status
Múltiples objetos enviados en mensaje multi-parte (protocolo MIME)	Cada objeto encapsulado en su propio mensaje de respuesta
Usa conexiones persistentes	Usa conexiones persistentes (HTTP 1.1)
Requiere que el mensaje completo (header y body) estén en ASCII	-
El servidor SMTP utiliza CRLF.CRLF para determinar el fin del mensaje.	Utiliza r\nr\n para indicar el fin del mensaje HTTP

Ejemplo de mensaje multi-parte MIME

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="--------517CA6F2DE4ABCAFCC5"
From: Lionel Messi <lmessi@afa.com.ar>
To: wweghorst@knvb.com
Subject: videito
Date: Fri, 09 Dec 2022 18:45:19 -0300

---------517CA6F2DE4ABCAFCC5
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit

Te dejo un video de los penales ;)

Anda pa'lla!

Leo

---------517CA6F2DE4ABCAFCC5
Content-Type: video/mp4
Content-Transfer-Encoding: base64
... bytes del video encodeados en base64 ...
```

Protocolos de acceso al Mail

- SMTP permite entregar y almacenar mensajes de e-mail

Los protocolos de acceso permiten obtener los mails de los servidores

- IMAP: Internet Mail Access Protocol [RFC 3501].
    - Provee recuperación de mails, borrado y administración de carpetas, entre otros (Los mensajes están almacenados en el servidor).

- HTTP
    - Gmail y Hotmail, por ejemplo, ofrecen una interfaz web sobre SMTP para enviar mensajes.
    - Gmail soporta IMAP (aunque el acceso a los mensajes vía web no se da por este protocolo)

El protocolo DNS: Domain Name System

Los dispositivos de Internet tienen:

- Una dirección IP (32 bits) —utilizada para enviar datagramas.
- Un nombre o *hostname*, ej., utdt.edu — utilizado por los humanos.

Domain Name System (DNS)

- Base de datos distribuida en una jerarquía de múltiples “servidores de nombre” (name servers)
- Protocolo de aplicación: los hosts se comunican con servidores DNS para resolver nombres (traducción entre nombres y direcciones)
- Funcionalidad central de Internet.

Ejemplo de cómo funciona

1. Un usuario abre su navegador web favorito y escribe la dirección http://utdt.edu en la barra de direcciones.
2. El navegador web extrae el nombre del host (hostname), utdt.edu, y se lo pasa a un cliente DNS.
3. El cliente DNS envía una consulta al servidor DNS que contiene el hostname requerido por el usuario.
4. El DNS eventualmente recibe una respuesta con la dirección IP del hostname
5. Una vez que el navegador recibe la dirección IP puede iniciar la conexión TCP con el servidor HTTP (puerto 80 en esa IP)

Servicios y estructura de DNS

Servicios de DNS

- Traducción de nombre: IP
- Host Aliasing. Un host con un hostname complicado puede tener uno o más nombres de alias. DNS puede ser invocado por una aplicación para obtener el hostname canónico a partir de un alias provisto o también la dirección IP de un host.
- Aliasing de servidores de mail. DNS puede ser invocado por una aplicación de mail para obtener el hostname canónico a partir del alias, así como también a partir de una dirección IP.
- Distribución de carga
    - Para servidores web replicados, muchas IPs se corresponden a un nombre.

¿Por qué no posee una estructura centralizada?

- Punto único de falla
- Volumen de tráfico
- Mantenimiento

Overview de como funciona DNS

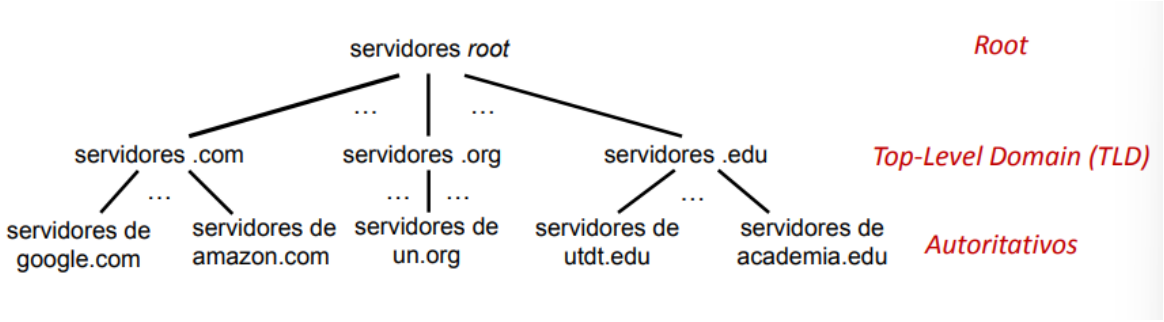
Supongamos que una aplicación esta corriendo en el host de un usuario y necesita traducir un hostname a una dirección IP. La aplicación va a invocar el lado del cliente de DNS, especificando el hostname que necesita que sea traducido. DNS toma el control y manda un mensaje query a la network. Todo los mensajes query de respuesta son mandados dentro de datagrama UDP al puerto 53. Después de un delay, el DNS dentro del host de usuario recibe un mensaje de respuesta DNS que provee el mapping deseado. Éste mapping es pasado a la aplicación que invocó.

Características de DNS

- Base de datos distribuida enorme
- Billones de consultas diarias
    - Gran volumen de lecturas vs. escrituras
    - Performance: toda transacción en Internet involucra DNS
- Descentralización organizacional y física

- Millones de organizaciones diferentes son responsables de sus propios registros.

Jerarquía de servidores DNS



Si un cliente quiere la dirección IP de www.utdt.edu,

1. Consulta un servidor root para encontrar un servidor TLD del dominio .edu
2. Consulta un servidor TLD de .edu para obtener un servidor autoritativo de utdt.edu
3. Consulta un servidor autoritativo de utdt.edu para obtener la IP de www.utdt.edu

Servidores TLD y autoritativos

Servidores Top-Level Domain (TLD)

- Existe un servidor TLD (o cluster de servidores) para cada dominio de nivel superior, incluyendo los de los países — .ar, .uk, .jp, et cetera.

Servidores autoritativos

- Servidores DNS propios de cada organización.
- Proveen traducciones de nombres a direcciones IP para los hosts de la organización.
- Pueden ser administrados por la misma organización o bien por un tercero (proveedor de servicios)

Servidores DNS locales

- Cuando un host realiza una consulta DNS, la misma se envía a su servidor DNS local

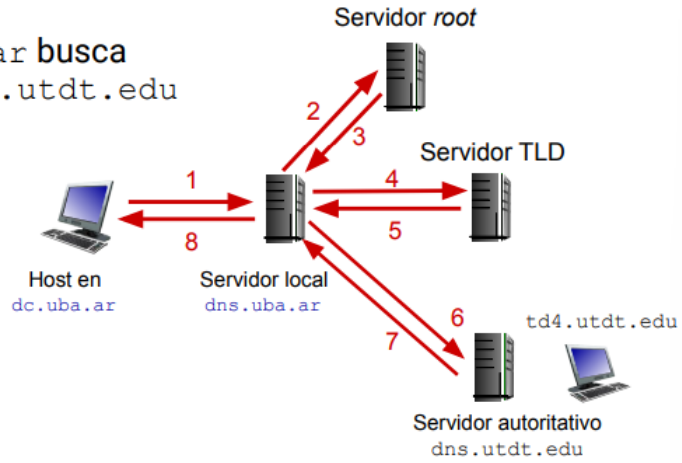
- La respuesta proviene de su caché local (podría estar desactualizada)
- También reenvía la consulta a la jerarquía DNS en caso de no tener la respuesta.

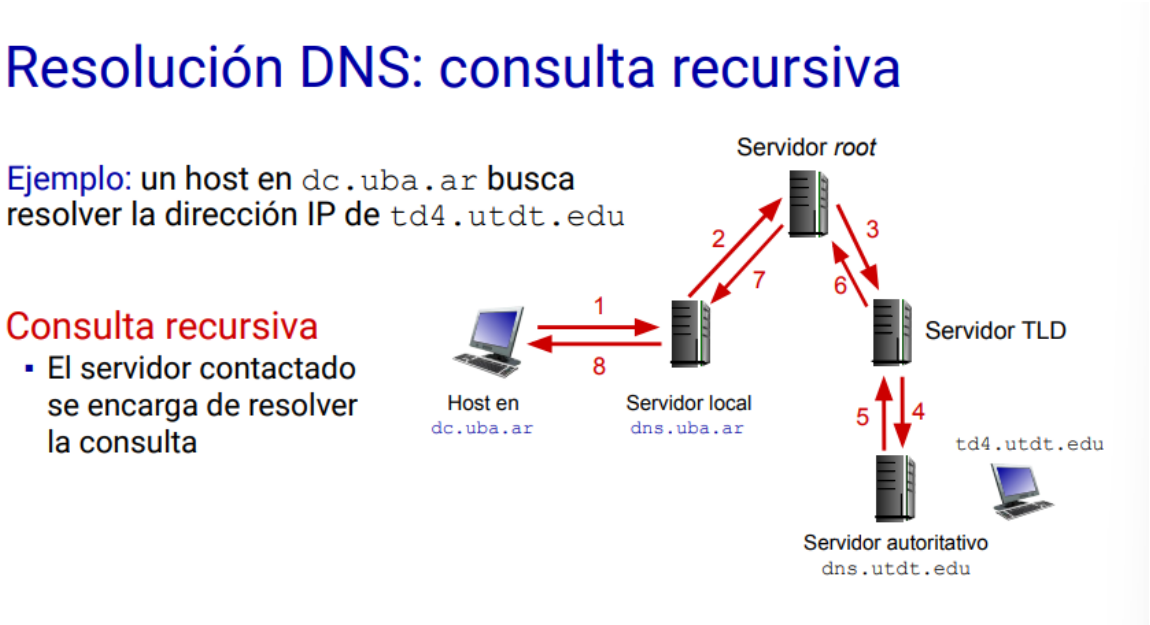
- Los ISPs tienen servidores DNS locales (se suele asignar uno automáticamente al host al conectarse)
- Notar que los servidores locales no pertenecen estrictamente a la jerarquía DNS

Resolución DNS: consulta iterativa

Ejemplo: un host en `dc.uba.ar` busca resolver la dirección IP de `td4.utdt.edu`

Consulta iterativa
- El servidor contactado responde con el nombre del próximo servidor a contactar
- *"No conozco este nombre, pero preguntale a este servidor"*





Caching en DNS

Cuando un servidor aprende un nuevo mapeo, lo cachea e inmediatamente devuelve este valor cacheado ante futuras consultas. Esto mejora los tiempos de respuesta. Sin embargo, los registros cacheados expiran luego de cierto tiempo dado a su TTL. Los servidores TLD suelen ser cacheados en los servidores locales.

Los registros en la caché pueden estar desactualizados. Si cierto host cambia su dirección IP, podría permanecer “oculto” hasta que todos los TTLs expiren. DNS ofrece una traducción “best-effort”

Registros DNS

DNS: base de datos distribuida que almacena registros (RR).

Formato: (nombre, valor, tipo, TTL)

1. Tipo A
    a. nombre: el host.
    b. valor: su dirección IP.
    c. Ejemplo (relay1.bar.foo.com, 147.37.93.126, A)
2. Tipo NS
    a. nombre: un dominio (e.g. utdt.edu).

b. valor: el hostname del servidor autoritativo para dicho dominio.
c. (foo.com, dns.foo.com, NS)

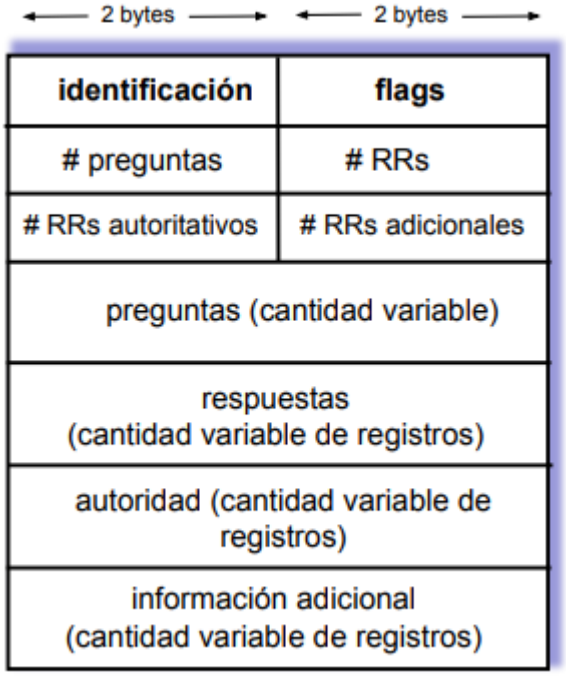
3. Tipo CNAME
   a. nombre: un alias para cierto nombre canónico (i.e., real)
   b. valor: el nombre canónico.
   c. (foo.com, relay1.bar.foo.com, CNAME)
4. Tipo MX
   a. El valor es el nombre de un servidor SMTP asociado con el nombre.
   b. (foo.com, mail.bar.foo.com, MX)

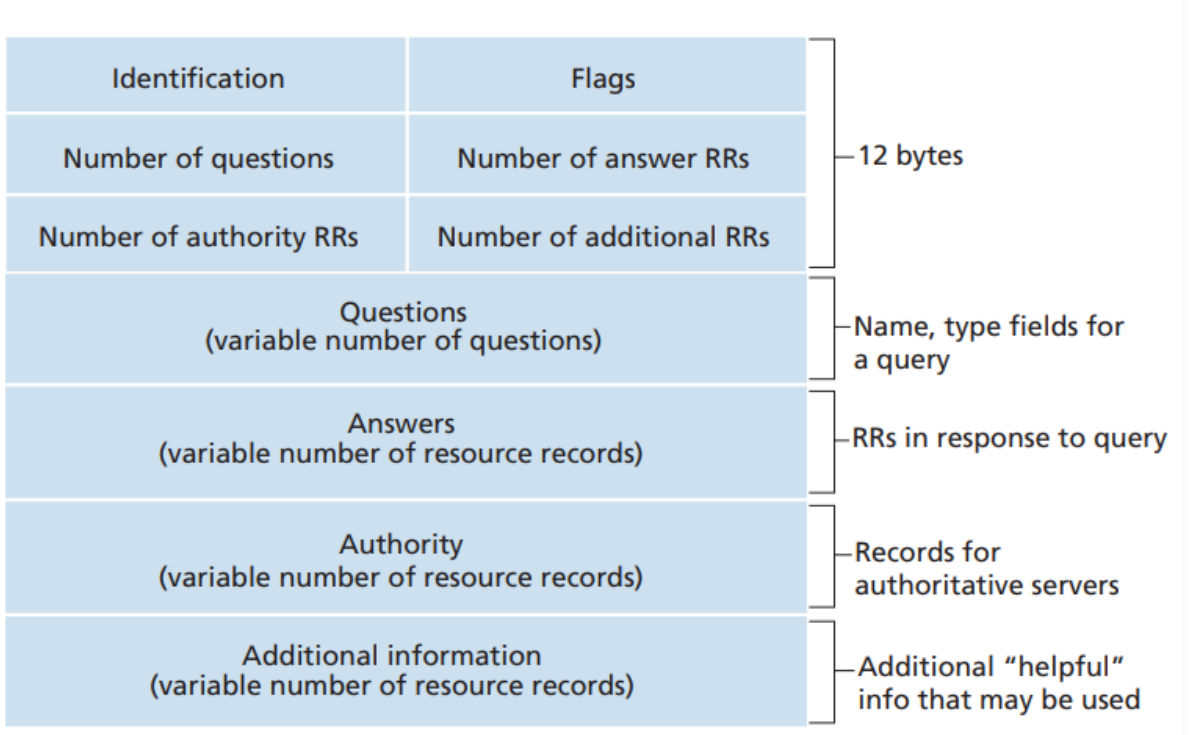
Mensaje DNS

Tanto las consultas (queries) como las respuestas (replies) tienen el mismo formato:

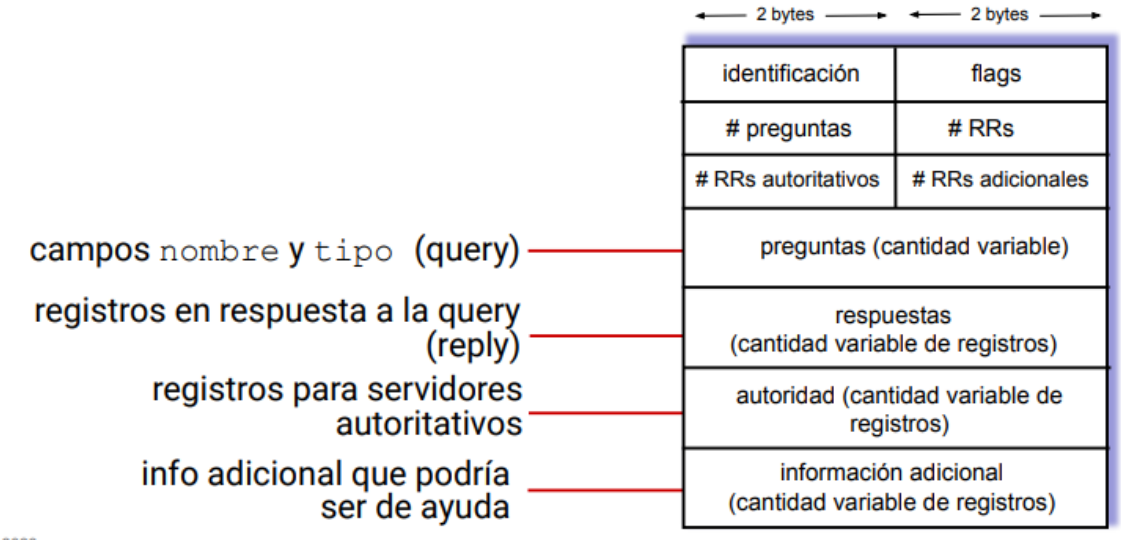
Header del mensaje:

- Identificación: número de 16 bits para la consulta (la respuesta utiliza el mismo)
- Flags:
  - query/reply
  - recursividad deseada
  - recursividad disponible
  - La respuesta es autoritativa.





Tanto las consultas (queries) como las respuestas (replies) tienen el mismo formato:



En una respuesta de un servidor DNS, la sección de respuesta contiene los Resource Records para el nombre por el que se consultó. Una respuesta puede devolver múltiples RRs en su mensaje, ya que un hostname puede tener múltiples direcciones IP.

La authority section contiene los Records de otros servidores autoritativos. Mientras que la additional section contiene otros Records útiles.

Nuevos registros en DNS

Supongamos que queremos dar de alta el dominio td4.com

1. Debemos registrar dicho nombre en un registrador DNS (e.g., Network Solutions)
    a. Proveemos nombres y direcciones IP de servidores autoritativos (primarios y secundarios).
    b. El registrador inserta NS y A en el servidor TLD de .com: (td4.com, dnsl.td4.com, NS) (dnsl.td4.com, 111.111.111.1, A)
2. Necesitamos generar un servidor DNS autoritativo con dirección IP 111.111.111.1
    a. Debe tener un registro A para www.td4.com
    b. Y, por ejemplo, otro de tipo MX (si queremos mails @td4.com)

Seguridad en DNS

Ataques DDoS

- Bombardeo de servidores root con tráfico:
  - No es muy factible
  - Filtrado de tráfico
  - Los servidores locales cachean las IPs de los servidores TLD (se bypassean lo servidores root)
- Bombardeo de servidores TLD
  - Potencialmente más peligroso

Ataques de spoofing

- Intercepción de queries para devolver respuestas falsas
  - DNS cache poisoning: las caches quedan envenenadas, direccionando tráfico a la máquina maliciosa.
  - RFC 4033: DNSSEC (ofrece servicios de autenticación)

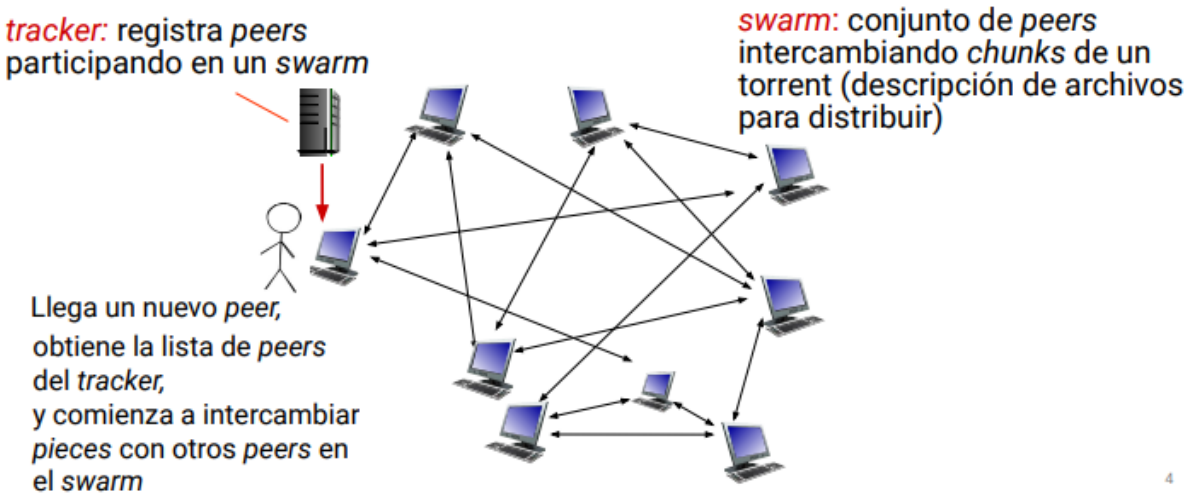
Aplicaciones P2P - BitTorrent

Arquitectura peer-to-peer (P2P)

- No hay un servidor
- Comunicación directa entre hosts/peers
- Los peers solicitan/ofrecen servicios a otros peers. (Autoescalabilidad)
- Los peers tienen conectividad intermitente; cambian direcciones IP

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



- 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 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 o permanecer en el torrent

Fórmula del tiempo mínimo de distribución:

$$D_{\text{P2P}} \geq \max \left\{ \frac{F}{u_s}, \frac{F}{d_{\min}}, \frac{NF}{u_s + \sum_{i=1}^{N} u_i} \right\}$$

Al principio de la distribución, solamente el servidor tiene el archivo. Por lo cual, el mínimo es al menos F/Us.

Mientras que el peer con el más bajo download rate no puede obtener todos los bits en menos de F/Dmin segundos.

Finalmente, la capacidad total de upload del sistema como un “todo” es igual al upload rate de cada peer individual; es decir, Utotal = Us + U1 + … + Un. Por último, el sistema debe subir F bits para cada N peers, entregando en total N*F bits.

N peers, NF bits, Upload rate Us y el tiempo que tarda en distribuir F/Dmin (Download rate)

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 (rares first—los más “Inusuales” primero). A veces, la política puede ser random first.

Tit-for-tat

Envío de chunks: tit-for-tat

Un peer encía chunks a los cuatro peers que le envían chunks a mayor velocidad. Los demás peers no reciben nada (chocked peers). El top 4 se reevalúa cada diez segundos.

Mientras que 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.

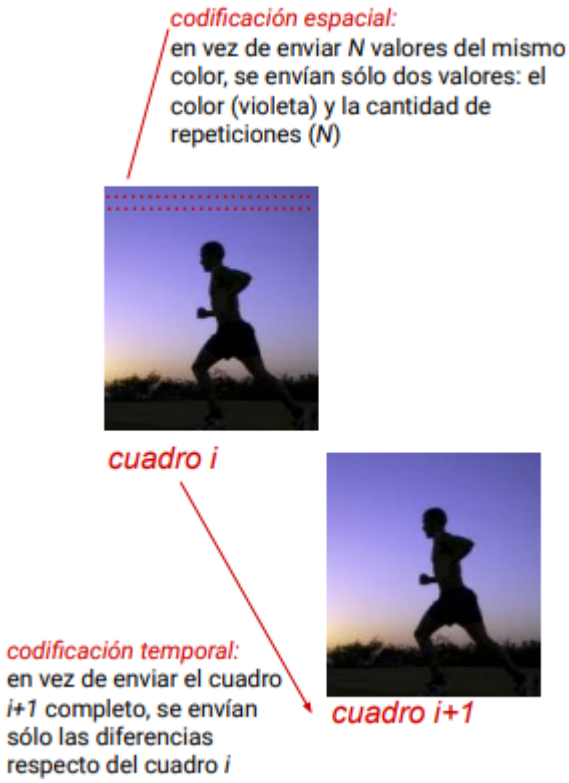
Video y compresión de video

El video es una secuencia de imágenes mostradas a una tasa constante. Mientras que una imagen (digital) es un arreglo de píxeles (cada píxel se representa con bits)

La codificación se trata de utilizar la redundancia de y entre imágenes para reducir la cantidad de bits del video. Hay dos tipos:

Espacial: dentro de la imagen

Temporal: de una imagen a la siguiente.

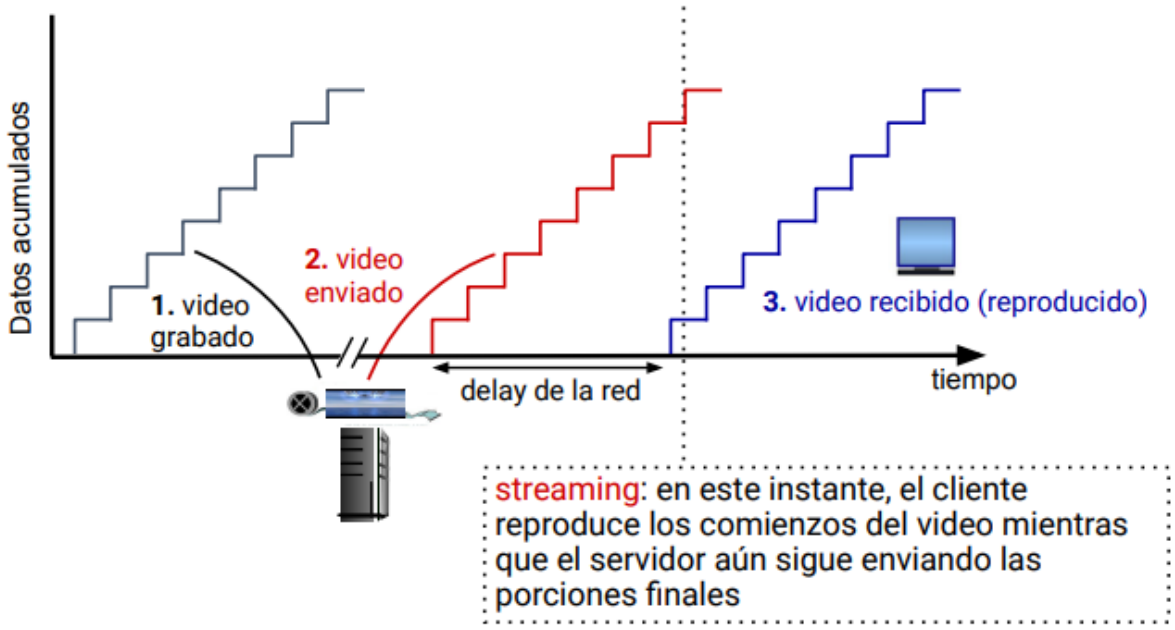


Dentro de la compresión de video nos encontramos con dos tipos:

- 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

Streaming de video almacenado

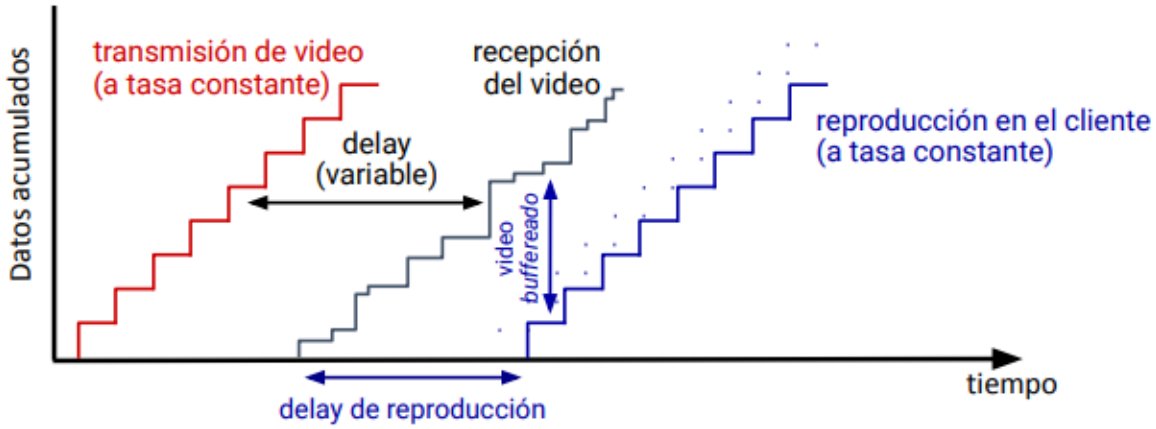
Los desafíos principales son que el ancho de banda del servidor al cliente puede cambiar a lo largo del tiempo (con niveles de congestión cambiantes) y que la pérdida de paquetes y el delay por congestión pueden demorar la reproducción del video o deteriorar su calidad.



Algunos desafíos del streaming en general incluyen:

- La 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 cliente. Esto hace que, del lado del cliente, los bytes son coleccionados en el buffer de la aplicación del cliente. Una vez el número de bytes en dicho buffer es excedido, la aplicación del cliente comienza el playback. Esto ocurre mientras la aplicación transmite un video y “bufferea” frames correspondientes a partes posteriores del vide.
- Interactividad. El cliente puede pausar, adelantar, rebobinar, et cetera.
- Pérdida de paquetes/retransmisiones

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 y calidades de reproducción.
- Estas codificaciones se almacenan en archivos distintos.
- Los archivos se replican en diversas CDNs.
- Un manifesto provee URLs para los distintos chunks.

Cliente

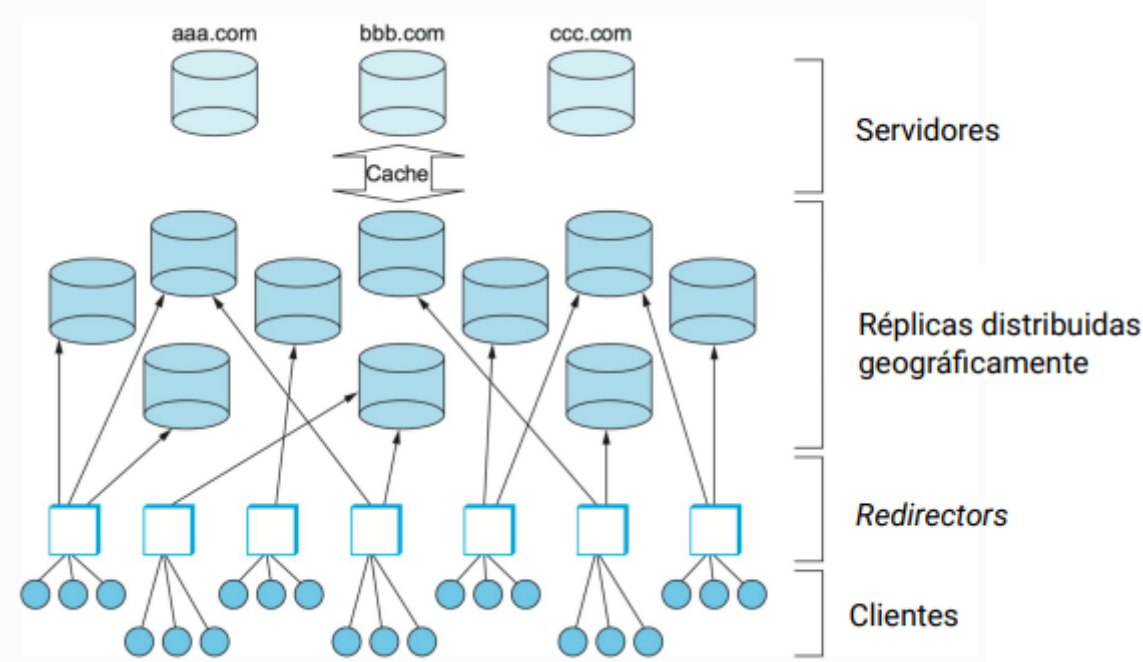
- Estima periódicamente el ancho de banda disponible desde el servidor.
- A través del manifesto, solicita un chunk por vez. Elige la máxima tasa de codificación sostenible dado el ancho de banda actual. Puede escoger diferentes tasas en diferentes instantes de tiempo (y también desde distintos servidores)
- “Inteligencia” en el cliwnte 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)
- DASH le permite al cliente adaptarse al ancho de banda disponible si hay cambios durante la sesión.

Redes de Distribución de Contenido (CDNs)

Es la manera en la que ofrecemos millones de videos a cientos de miles de usuarios simultáneos.

Los CDNs almacenan múltiples copias de los videos en diversos sitios geográficamente distribuidos.



Las CDNs almacenan copias de contenido multimedia en sus nodos. Los usuarios, al solicitar el contenido, reciben un manifiesto del proveedor. Mediante éste, el cliente puede obtener el contenido a la mayor tasa de transferencia soportada. Puede elegir una tasa o una copia distinta. Sin embargo, el CDN no va a querer tener una copia del mismo video en cada cluster ya sea por popularidad o por otro motivo. Entonces cuando un cliente pide un video específico, el cluster recupera el video y lo almacena localmente mientras el cliente mira el video al mismo tiempo. Similar a Web Caching. El CDn debe interceptar el request así puede determinar el cluster del servidor CDN apropiado para el cliente en el momento y redirigir el

request del cliente hacia el servidor en dicho cluster. Para elegir el cluster apropiado hasy dos estrategias. La más simple es asignarle al cliente al cluster más geográficamente cerca. Pero esto podría no funcionar para todos ya que tal vez el cluster más “cercano” puede no ser el más cercano en términos de “saltos” en la red. Usualmente, se opta por hacer mediciones en tiempo real del delay y pérdida de performance entre clusters y clientes para determinar cual es el más apropiado.

Para la organización de CDNs hay dos filosofías:

1. Enter Deep. Adentrarse profundamente a las networks de acceso de los Internet Service Providers, mediante la implementación de clusters de servers en puntos de acceso a ISPs en todo el mundo. La meta es acercarse a los end users y así mejorar el delay de la perspectiva de usuario y el throughput a través de la disminución de número de links y routers entre el end user y el servidor CDN del cual recibe su contenido. Sin embargo, el mantenimiento puede ser complicado.
2. Bring Home. Es traer a los ISPs *home* o a casa al construir grandes cluster en un menor número de sitios. En vez de adentrarse en el acceso del ISPs, los CDNs son colocados en los Internet Exchange Points (IXP). Este diseño resulta en un menor mantenimiento.