

Tecnología Digital IV: Redes de Computadoras

Clase 5: Nivel de Aplicación - Parte 3

Lucio Santi & Emmanuel Iarussi

Licenciatura en Tecnología Digital
Universidad Torcuato Di Tella

25 de marzo de 2025



Agenda

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

E-mail y el protocolo SMTP

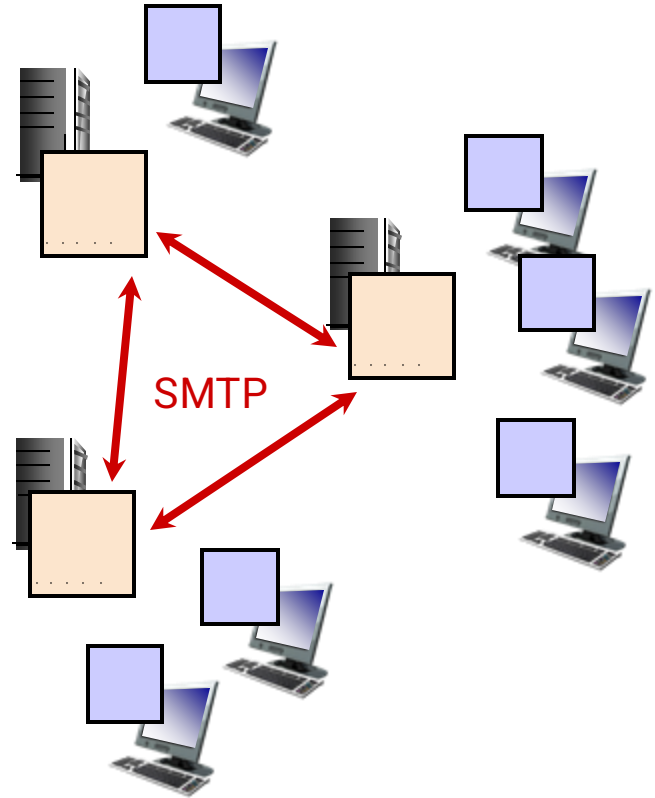
E-mail

Tres componentes principales:

- *User agents* 
- Servidores de mail 
- *Simple Mail Transfer Protocol*: **SMTP**

User Agent

- Permite redactar, editar y leer mensajes
- e.g., Outlook
- Los mensajes se almacenan en el servidor



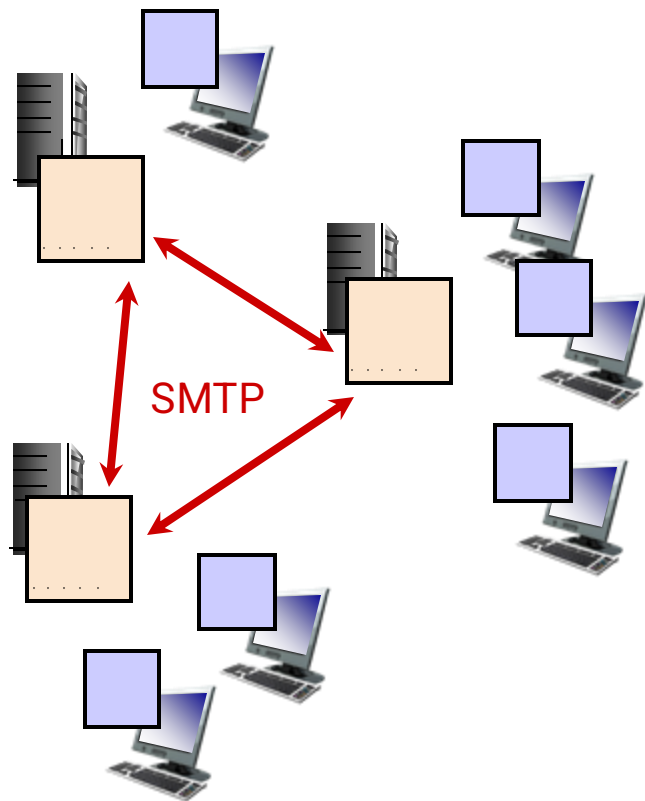
Servidores de mail

En los servidores tenemos:

- El **buzón** (*mailbox*) que guarda los mensajes entrantes para el usuario
- Una **cola de salida** de los mensajes a ser enviados

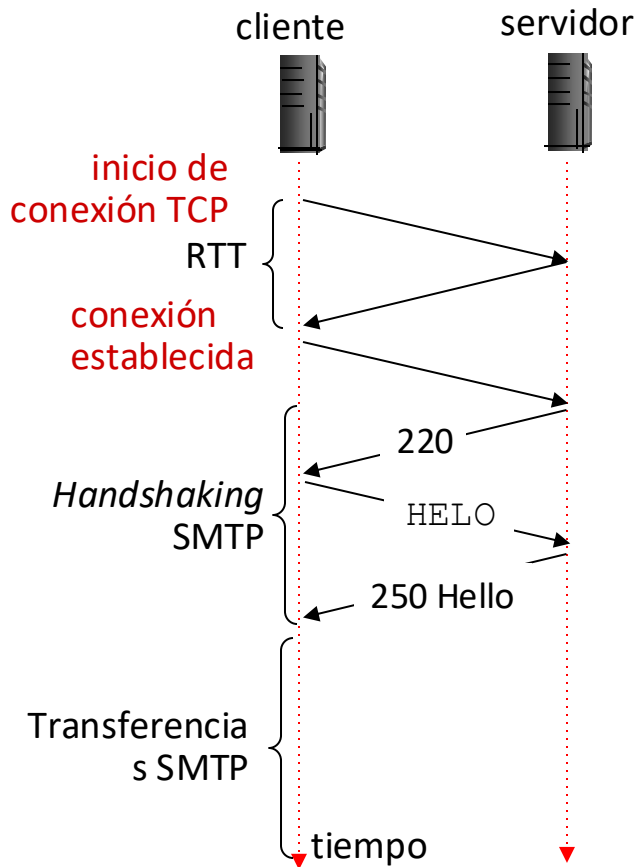
El protocolo **SMTP** permite enviar mensajes entre servidores

- **Cliente:** el servidor de mail emisor
- **Servidor:** el servidor de mail receptor



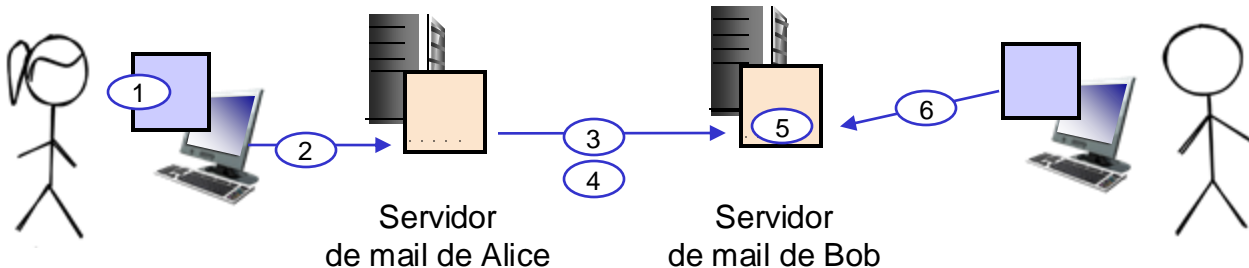
SMTP

- Utiliza TCP para transferencias confiables (el servidor escucha en el puerto 25)
- Tres fases:
 - *Handshaking*
 - Transferencia de mensajes
 - Cierre
- Interacciones de tipo comando/respuesta (como HTTP)
 - Formato en texto (ASCII)



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



Ejemplo: interacción SMTP

S: 220 utdt.edu

C: HELO gmail.com

itse

S: 221 utdt.edu closing connection

Ejemplo: interacción SMTP

S: 220 utdt.edu

C: HELO gmail.com

S: 250 Hello gmail.com, pleased to meet you

C: MAIL FROM: <usuarioA@gmail.com>

itse

S: 221 utdt.edu closing connection

Ejemplo: interacción SMTP

```
S: 220 utdt.edu
C: HELO gmail.com
S: 250 Hello gmail.com, pleased to meet you
C: MAIL FROM: <usuarioA@gmail.com>
S: 250 usuarioA@gmail.com... Sender ok
C: RCPT TO: <usuarioB@utdt.edu>
```

itse

```
S: 221 utdt.edu closing connection
```

Ejemplo: interacción SMTP

```
S: 220 utdt.edu
C: HELO gmail.com
S: 250 Hello gmail.com, pleased to meet you
C: MAIL FROM: <usuarioA@gmail.com>
S: 250 usuarioA@gmail.com... Sender ok
C: RCPT TO: <usuarioB@utdt.edu>
S: 250 usuarioB@utdt.edu ... Recipient ok
C: DATA
```

itse

```
S: 221 utdt.edu closing connection
```

Ejemplo: interacción SMTP

```
S: 220 utdt.edu
C: HELO gmail.com
S: 250 Hello gmail.com, pleased to meet you
C: MAIL FROM: <usuarioA@gmail.com>
S: 250 usuarioA@gmail.com... Sender ok
C: RCPT TO: <usuarioB@utdt.edu>
S: 250 usuarioB@utdt.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by
itself
C: Hola! Vas a la clase de TD4 hoy?
C: Saludos!
```

```
S: 221 utdt.edu closing connection
```

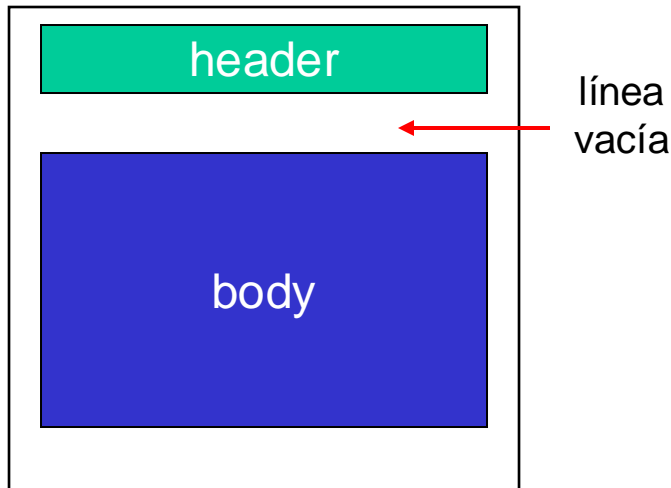
Formato de los mensajes

- líneas de headers, e.g.

- To:
- From:
- Subject:

Notar que estos headers son diferentes a los comandos SMTP `MAIL FROM:`, `RCPT TO:`!

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



La sintaxis de los mensajes de e-mail se define en el RFC 2822

Demo!

- Utilicemos **netcat** para conectarnos con un servidor de mail y hablar SMTP a través del teclado
- Ejecutar
 - `nc -v mail.guerrillamail.com 25`
- Enviar un e-mail de la dirección `test@hotmail.com` a `test_td4@guerrillamail.com`
- Ingresar a www.guerrillamail.com y comprobar que el mensaje se recibió correctamente en el *mailbox* del usuario indicado

Demo!

```
[
Connection to mail.guerrillamail.com port 25 [tcp/smtp] succeeded!
220 mail.guerrillamail.com SMTP Guerrilla(v1.6.1) #115823533 (175) 2024-08-19T15:55:01Z
[HELO hotmail.com
250 mail.guerrillamail.com Hello
MAIL FROM: <test@hotmail.com>
250 2.1.0 OK
RCPT TO: <test_td4@guerrillamail.com>
250 2.1.5 OK
DATA
354 Enter message, ending with '.' on a line by itself
PRUEBA
TEXT0 TD4
.
250 2.0.0 OK: queued as 860142160126f715a58726ed15166fb0
QUIT
221 2.0.0 Bye
marceloromeo@MacBook-Air-de-Marcelo ~ % █
```

Demo

Guerrilla Mail - Disposable Temporary E-Mail Address

Avoid spam and stay safe - use a disposable email address! Click the "WTF" button below for help. So far we've processed **16,977,099,817 emails (+485)**, Keeping your real inbox safe and clean (58899 emails going in / hour)

test_td4 @ guerrillamail.com [Forget Me](#) [WTF?](#)

tze4y5+5l3ugk57mlgfc@guerrillamail.com ☒ [Scramble Address](#)

EMAIL

COMPOSE

TOOLS

ABOUT

« Back to inbox

Reply

Forward

Show Original

From: test@hotmail.com, To: test_td4, Date 2024-08-19 15:56:01

Delivered-To: test_td4@sharklasers.com
Received: from 181.165.102.27 ([181.165.102.27])
by guerrillamail.com with SMTP id
860142160126f715a58726ed15166fb0@guerrillamail.com;
Mon, 19 Aug 2024 15:55:53 +0000

PRUEBA
TEXT0 TD4

SMTP vs. HTTP

- HTTP: *pull* del cliente
- SMTP: *push* del cliente
- Ambos tienen interacciones comando/respuesta en ASCII y códigos de status
- HTTP: cada objeto encapsulado en su propio mensaje de respuesta
- SMTP: múltiples objetos enviados en mensaje *multiparte* (protocolo MIME)
- SMTP usa conexiones persistentes
- SMTP requiere que el mensaje completo (header y body) estén en ASCII
- El servidor SMTP utiliza CRLF.CRLF para determinar el fin del mensaje

Ejemplo de mensaje multipart MIME

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="-----517CA6F2DE4ABCAFFCC5"
From: Lionel Messi <lmessi@afa.com.ar>
To: wweghorst@knvb.com
Subject: videito
Date: Fri, 09 Dec 2022 18:45:19 -0300
```

```
-----517CA6F2DE4ABCAFFCC5
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
```

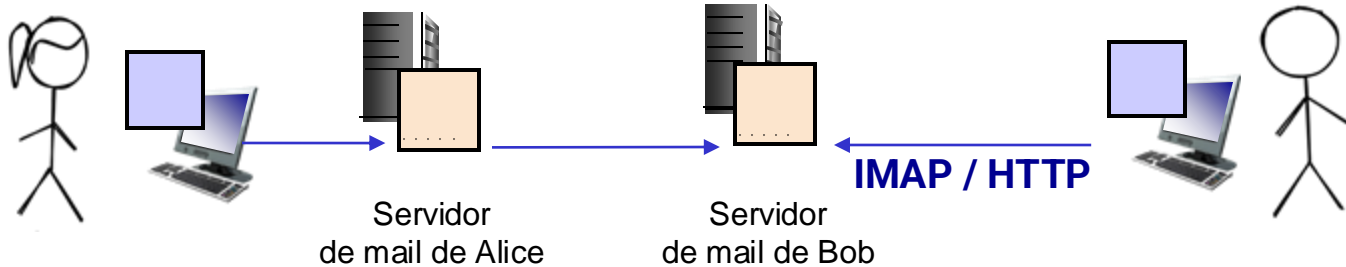
Te dejo un video de los penales ;)

Anda pa'lla!

Leo

```
-----517CA6F2DE4ABCAFFCC5
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

DNS: *Domain Name System*

Los dispositivos en Internet tienen:

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

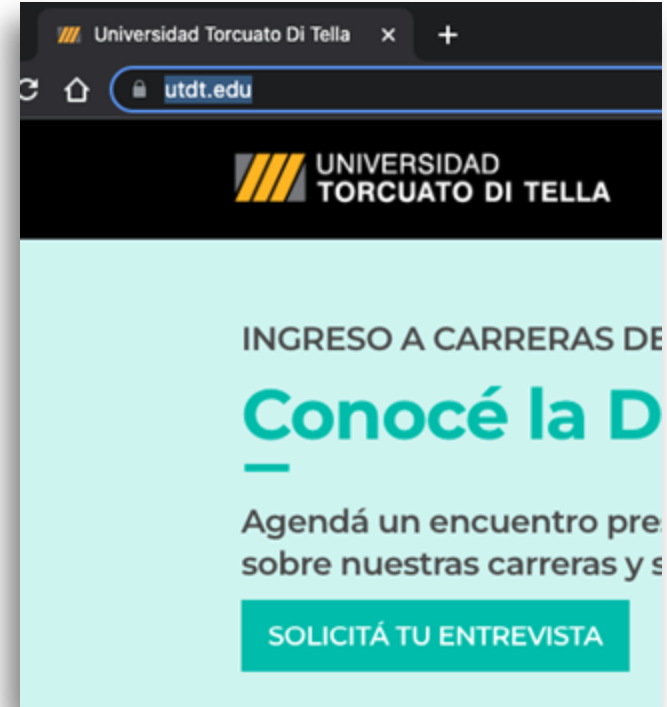
¿Cómo podemos mapear una IP a un nombre y viceversa?

Domain Name System (DNS)

- **Base de datos distribuida** en una jerarquía de múltiples “servidores de nombres” (*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 nombre → IP
- *Aliasing*
 - Nombres canónicos y alias
- Aliasing de servidores de mail
- Distribución de carga
 - Para servidores web replicados, muchas IPs se corresponden a un nombre

¿Por qué no centralizarlo?

- Punto único de falla
- Volumen de tráfico
- Mantenimiento

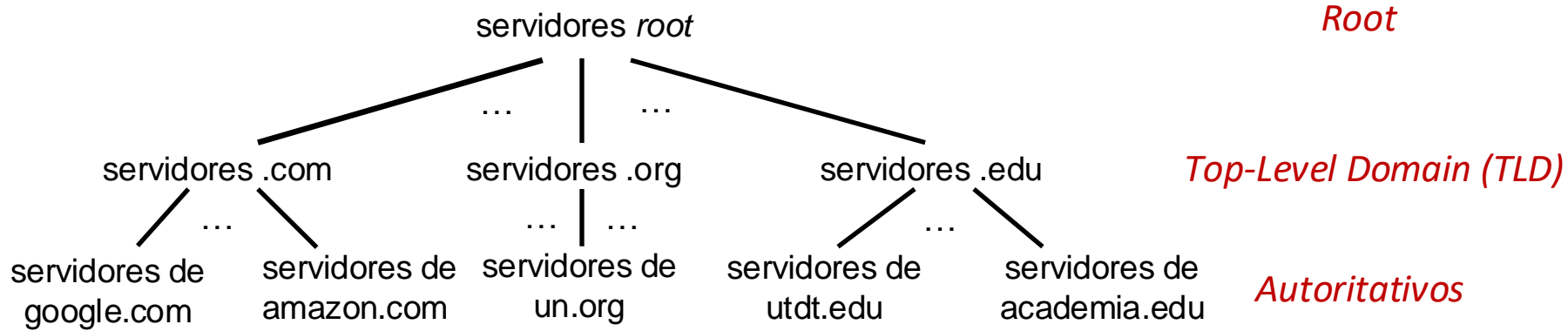
En definitiva, **no es escalable**

- Sólo los servidores DNS de Akamai reciben ~2.2B queries DNS por día

Características de DNS

- Base de datos distribuida **enorme**
 - Miles de millones de registros
- **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

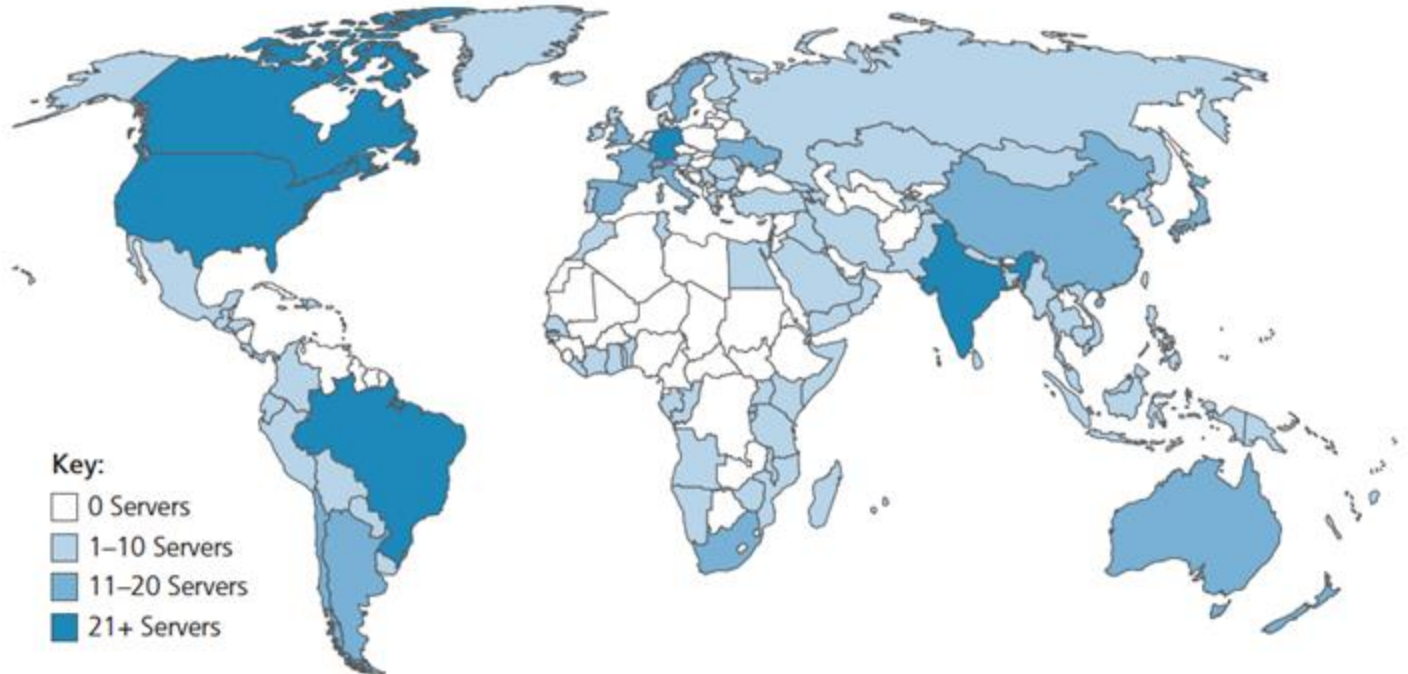


Si un cliente quiere la dirección IP de `www.utdt.edu`,

- Consulta un servidor *root* para encontrar un servidor TLD del dominio `.edu`
- Consulta un servidor TLD de `.edu` para obtener un servidor autoritativo de `utdt.edu`
- Consulta un servidor autoritativo de `utdt.edu` para obtener la IP de `www.utdt.edu`

Servidores *root*

Hay 13 servidores root distintos con más de 1000 réplicas a lo largo de todo el mundo:



Servidores TLD y autoritativos

Servidores *Top-Level Domain* (TLD)

- Existe un servidor TLD (o cluster de servidores) para cada dominio de nivel superior (e.g., .com, .org, .net, .edu, .aero, .jobs, .museums), incluyendo los de los países – .ar, .uk, .fr, .jp, etc.
- Por ejemplo, la compañía Educause administra los servidores DNS del dominio .edu

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

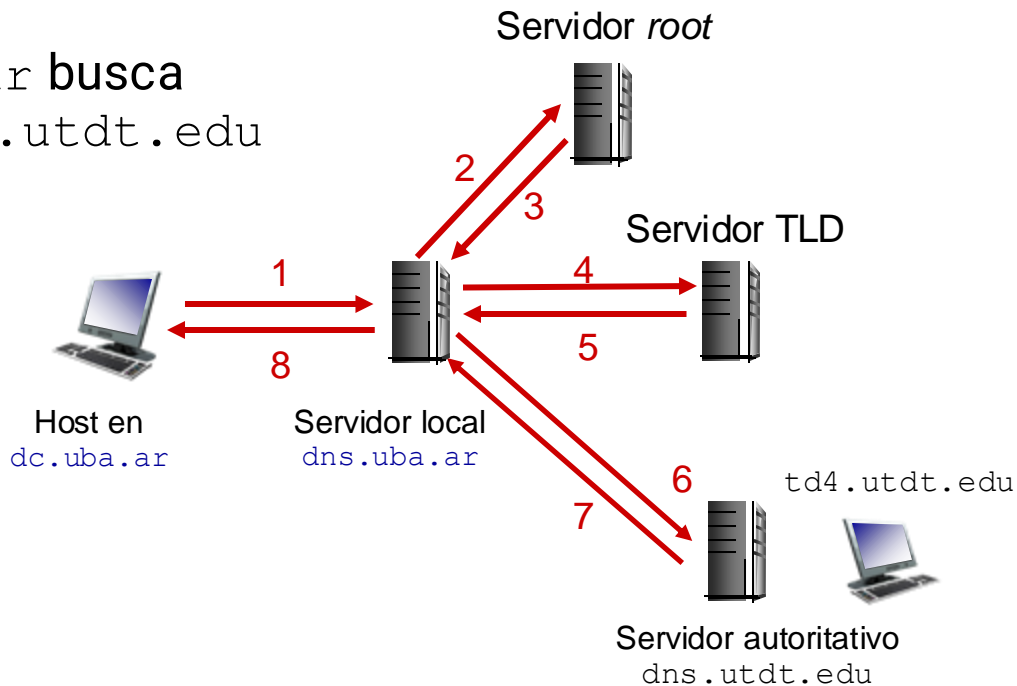
- Cuando un host realiza una consulta DNS, la misma se envía a su servidor DNS **local**
 - La respuesta proviene de su caché local (podría estar desactualizada)
 - También reenvía la consulta a la jerarquía DNS en caso de no tener la respuesta
- Los ISPs tienen servidores DNS locales (se suele asignar uno automáticamente al host al conectarse)
- Notar que estos 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”*

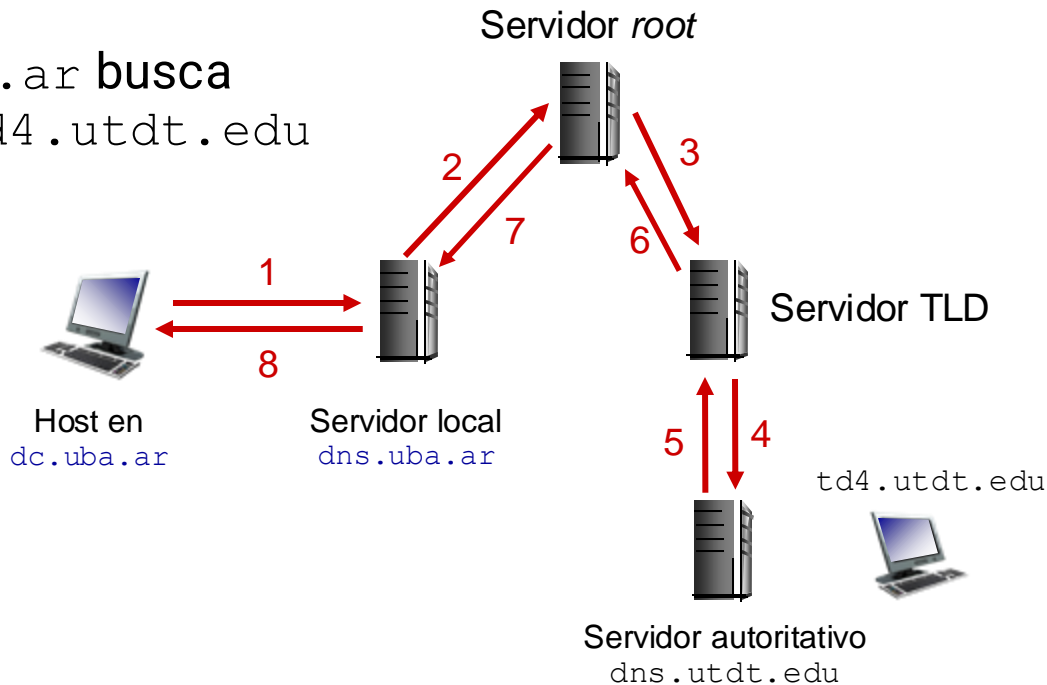


Resolución DNS: consulta recursiva

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

Consulta recursiva

- El servidor contactado se encarga de resolver la consulta



Caching en DNS

- Cuando un servidor aprende un nuevo mapeo, lo **cachea** e inmediatamente devuelve este valor cacheado ante futuras consultas (aún si no es su servidor autoritativo)
 - Mejora los tiempos de respuesta
 - Los registros cacheados expiran luego de cierto tiempo (*Time To Live*, **TTL**)
 - Los servidores TLD suelen estar 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)

tipo **A**

- nombre: el host
- valor: su dirección IP

tipo **NS**

- nombre: un dominio (e.g. utdt.edu)
- valor: el hostname del servidor autoritativo para dicho dominio

tipo **CNAME**

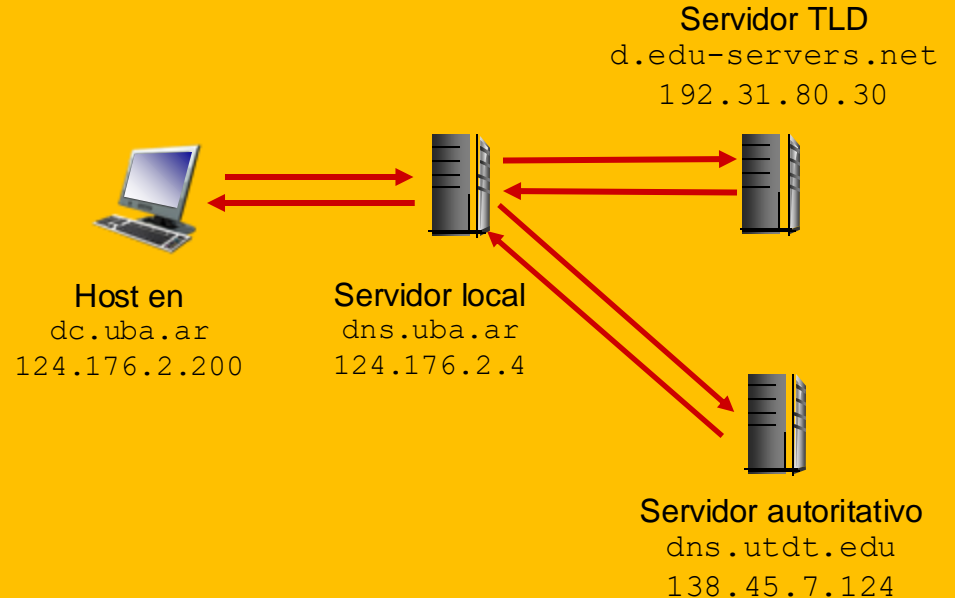
- nombre: un *alias* para cierto nombre *canónico* (i.e., real)
- valor: el nombre canónico

tipo **MX**

- El `valor` es el nombre de un servidor SMTP asociado con el nombre

Ejercicio!

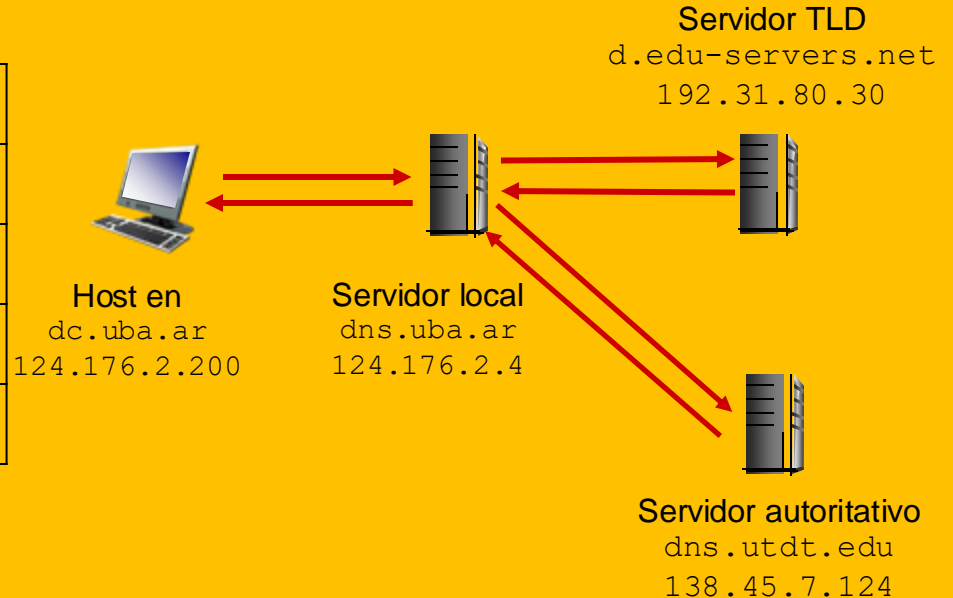
- Supongamos que el servidor TLD para dominios `.edu` no es autoritativo para el host `utdt.edu`. ¿Qué registros DNS contendrá el servidor TLD? ¿De qué tipo?



Ejercicio!

- Supongamos que el servidor TLD para dominios `.edu` no es autoritativo para el host `utdt.edu`. ¿Qué registros DNS contendrá el servidor TLD? ¿De qué tipo?

Tipo	Name	Value
...
NS	utdt.edu	dns.utdt.edu
A	dns.utdt.edu	138.45.7.124
...



Mensajes DNS

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

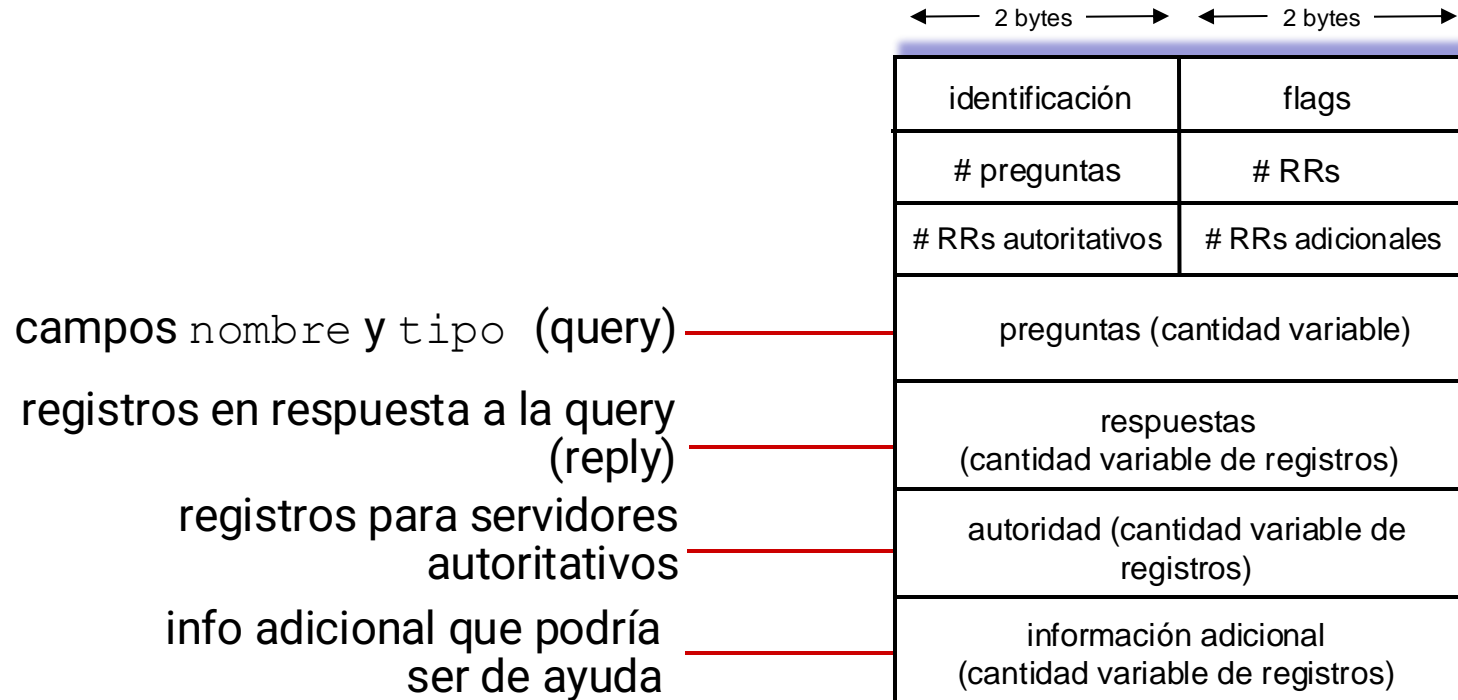
Header del mensaje:

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



Mensajes DNS

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



Nuevos registros en DNS

Supongamos que queremos dar de alta el dominio `td4.com`

- Debemos registrar dicho nombre en un **registrador DNS** (e.g., [Network Solutions](#))
 - Proveemos nombres y direcciones IP de servidores DNS autoritativos (primarios y secundarios)
 - El registrador inserta registros NS y A en el servidor TLD de `.com`:
(`td4.com`, `dns1.td4.com`, NS)
(`dns1.td4.com`, `111.111.111.1`, A)
- Necesitamos generar un servidor DNS autoritativo con dirección IP `111.111.111.1`
 - Debe tener un registro A para `www.td4.com`
 - 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* los 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)