



Módulo 02

La Capa de Aplicaciones

(Pt. 3)



Redes de Computadoras
Depto. de Cs. e Ing. de la Comp.
Universidad Nacional del Sur



Copyright

- Copyright © **2010-2024** A. G. Stankevicius
- Se asegura la libertad para copiar, distribuir y modificar este documento de acuerdo a los términos de la **GNU Free Documentation License**, versión 1.2 o cualquiera posterior publicada por la Free Software Foundation, sin secciones invariantes ni textos de cubierta delantera o trasera
- Una copia de esta licencia está siempre disponible en la página <http://www.gnu.org/copyleft/fdl.html>
- La versión transparente de este documento puede ser obtenida de la siguiente dirección:

<http://cs.uns.edu.ar/~ags/teaching>



Contenidos

- Servicios que requiere la capa de aplicaciones
- Protocolos de la capa de aplicaciones
 - HTTP
 - DNS
 - SMTP, POP e IMAP
- Arquitectura de las aplicaciones P2P
- Programación basada en sockets



Directorio de nombres

- Uno de los requerimientos de las aplicaciones de red es poder identificar unívocamente a cualquier computadora dentro de la red
 - ➔ A nivel de hardware se usa un patrón de 32 bits denominado **dirección IP**
 - ➔ Para facilitar la comunicación de direcciones **IP** se suele cortar este patrón en cuatro partes de 8 bits, expresadas en decimal y separadas por un punto
- Aún con esa convención, recordar múltiples direcciones **IP** se vuelve bastante complicado...



Directorio de nombres

- Los humanos sin duda retenemos con más facilidad nombres simbólicos concretos
 - Por caso, **cs.uns.edu.ar**
- Necesitamos un **servicio** que nos permita convertir nombre simbólicos en direcciones **IP**
- Características que debería tener este servicio:
 - El sistema de directorio de nombres de internet necesita ser **confiable** y **eficiente**
 - La base de datos resultante es **altamente dinámica**, ya que se producen altas y bajas constantemente



Sistema centralizado

- La solución obvia sería ensayar alguna forma de **sistema centralizado**
 - Por caso, el archivo **/etc/hosts** de los sistemas **UNIX**
- No obstante:
 - Poca tolerancia a fallos
 - Volumen de tráfico astronómico
 - Imposible estar cerca de todas partes a la vez
 - Imposible mantener al día la base de datos
 - Alto costo para escalar el sistema



Domain Name System

- La solución adoptada para este complicado problema es hacer uso de una base de datos distribuida junto con el correspondiente protocolo de la capa de aplicaciones
 - Se denomina **DNS** (Domain Name System)
 - La base de datos distribuida está compuesta por una **jerarquía de servidores de dominio**
 - Computadoras, routers y servidores de dominio se comunican entre sí usando el **protocolo DNS**
 - La complejidad se lleva a la **frontera de la red**



Domain Name System

● Rol central:

- Mapear nombres simbólicos a direcciones **IP**

● Roles secundarios:

- Implementar **sinónimos** simbólicos. Nótese que para esto hace falta distinguir a uno de los sinónimos como el **nombre canónico**
- Usar estos sinónimos en los servidores de correo
- Posibilitar un esquema de **distribución de carga** simple pero básico: asociando las direcciones **IP** de múltiples servidores a un mismo nombre simbólico

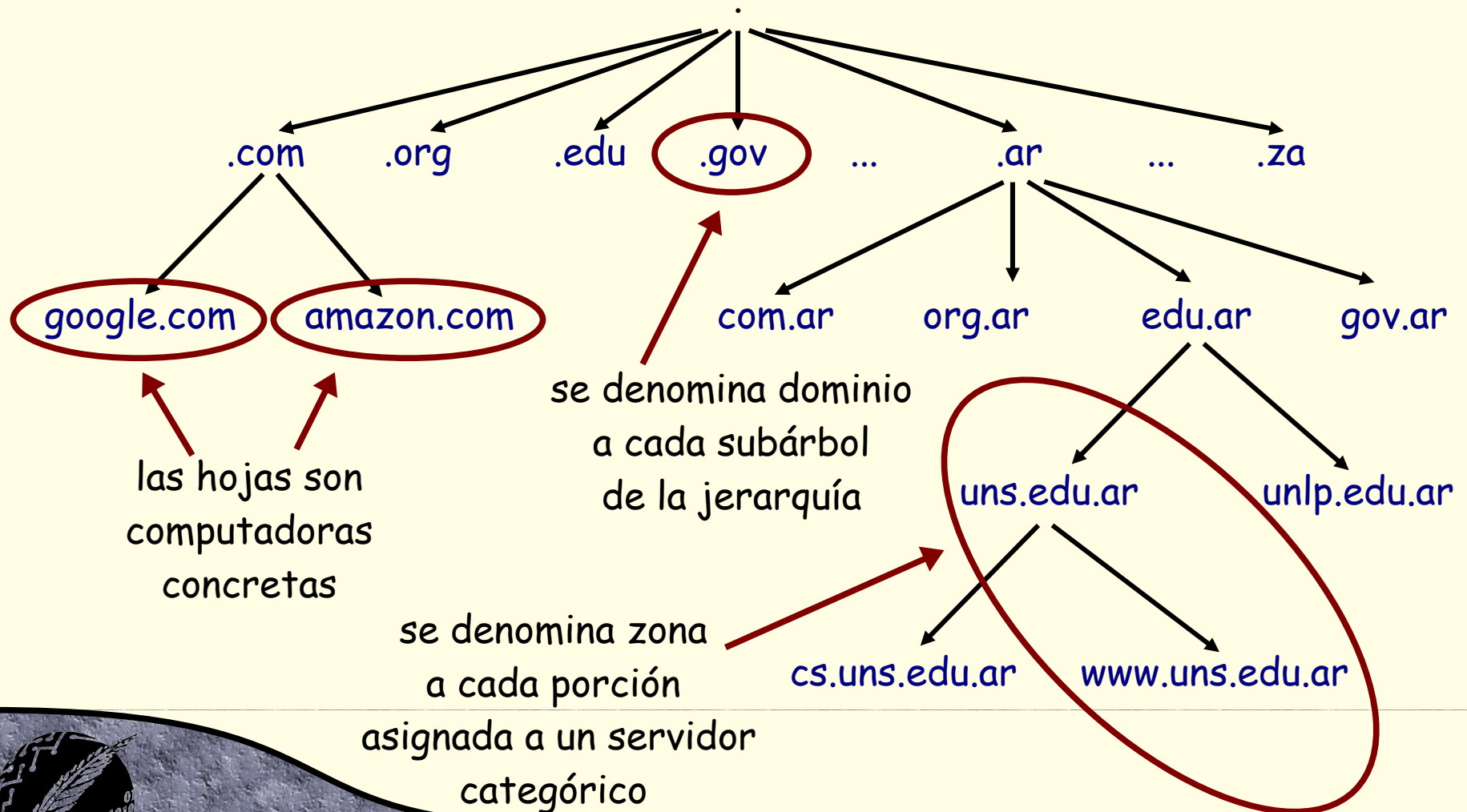


Domain Name System

- Ningún servidor conoce la totalidad del mapeo bajo este esquema distribuido
 - El servidor de dominio se denomina **categórico** cuando conoce de primera mano el mapeo para un determinado conjunto de computadoras
 - Los servidores, categóricos o no, **se organizan jerárquicamente** en función de la cantidad de computadoras a su cargo
 - Los servidores en el tope de la jerarquía se denominan **servidores raíz**



Jerarquía de servidores



Servidores raíz

- Hay 13 servidores raíz en todo el mundo...



...pero en realidad ¡hay más de 600!



Entidades a cargo

e NASA Mt View, CA

f Internet Software C., Palo Alto, CA (y otros 48 lugares)

k RIPE Londres (y otros 17 lugares)

i Netnod, Estocolmo (y otros 37 lugares)

m WIDE, Tokio (y otros 5 lugares)

a Verisign, Los Angeles, CA (y otros 5 lugares)

b USC-ISI, Marina del Rey, CA

l ICANN, Los Angeles, CA (y otros 41 lugares)

c Cogent, Herndon, VA (y en otros 5 lugares)

d U Maryland College Park, MD

g US DoD Columbus, OH (y en otros 5 lugares)

h ARL Aberdeen, MD

j Verisign, Dulles, VA (y en otros 69 lugares)



DNS en la actualidad

- La carga en los servidores raíz creció de gran forma con el advenimiento de la web
- Los **dominios de primer nivel** (top-level domain) se solían clasificar en dos grandes categorías:
 - Por un lado están los **dominios de primer nivel genéricos** (**.com**, **.edu**, **.gov**, **.org**, **.mil** y **.net**)
 - Por otra parte están los **dominios de primer nivel de países** (por caso, **.ar**, **.br**, **.cl**, etc.)
- En agosto de 2000 los servidores raíz pudieron delegar los dominios de primer nivel genéricos



DNS en la actualidad

- En la actualidad los dominios de primer nivel se clasifican en múltiples categorías
- Dominios sin una organización detrás:
 - ➔ **.com**, **.edu**, **.gov**, **.org**, **.mil** y **.net**
 - ➔ **.biz** (para las empresas)
 - ➔ **.info** (información en general)
 - ➔ **.name** (para personas individuales)



DNS en la actualidad

● Dominios con una organización detrás:

- ➔ **.aero** (industria aeronáutica)
- ➔ **.coop** (cooperativas)
- ➔ **.museum** (museos)
- ➔ **.travel** (industria del turismo)

● Dominios nuevos más recientes:

- ➔ **.mobi** (dominios optimizados para celulares)
- ➔ **.tel** (relativo a la industria telefónica)



DNS en la actualidad

- La situación en Argentina es muy buena, la gestión de los dominios **.ar** está a cargo del sitio **nic.ar** que depende del gobierno
 - ➔ Hasta el año 2013 el registro y la delegación de los dominios **.com.ar** era gratuita, una excelente medida que propendía a fomentar la adopción de estas tecnologías
 - ➔ En la actualidad si bien se cobra un costo fijo por la mayoría de los trámites, el mismo sigue siendo accesible pues aumenta menos que la inflación
 - Por caso, **\$855** por un alta de dominio



Domain Name System

- Cada red local de computadoras cuenta con un **servidor de dominio por defecto**
 - Las computadoras de la red acceden en primer lugar al servidor por defecto
 - Si el servidor local no fuera capaz de resolver una consulta, se accederá a uno de los servidores raíz
 - El servidor raíz contacta al servidor categórico correspondiente, a fin de poder acercarle al servidor local el mapeo que no había podido resolver
 - Cada zona cuenta con su respectivo conjunto de servidores categóricos (principal y secundario)

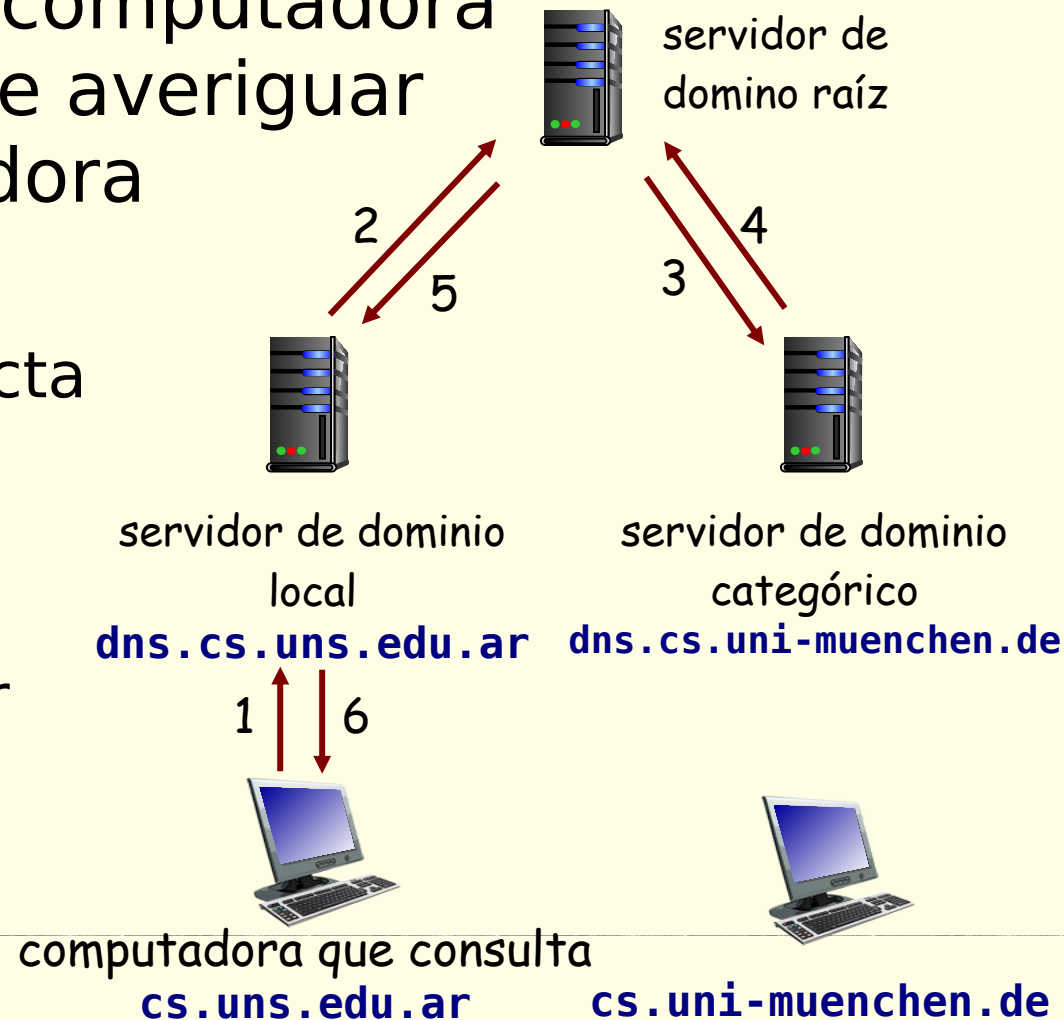


Ejemplo de consulta DNS

Supongamos que la computadora **cs.uns.edu.ar** quiere averiguar el **IP** de la computadora **cs.uni-muenchen.de**

→ En primer lugar contacta a **dns.cs.uns.edu.ar**, su servidor **DNS** local

→ **dns.cs.uns.edu.ar** contacta a un servidor raíz, de ser necesario



Ejemplo de consulta DNS

- El servidor raíz se contacta con el servidor categórico, de ser necesario
- El servidor raíz es posible que no conozca quién es el servidor categórico
- No obstante, si es usual que conozca a quién preguntar al respecto
- El nodo padre almacena esa información



Consultas iteradas

- El servidor raíz tiene dos opciones a la hora de contestar a una consulta que no haya podido resolver apelando a sus propios registros:
 - ➔ Por un lado puede encargarse él mismo de averiguar cómo responder a la consulta. Esta modalidad se denomina **resolución recursiva de consultas**
 - ➔ Otra alternativa es contestar simplemente “no lo sé, pero este servidor quizás si lo sepa”. Esta modalidad se denomina **resolución iterada de consultas**



Picando en el área chica...



You

necesito entender la diferencia entre la resolución recursiva y la resolución iterada de las consultas a través del protocolo DNS



ChatGPT

La diferencia entre la resolución recursiva y la resolución iterada en el protocolo DNS se refiere a cómo se realizan las consultas de nombres de dominio y se obtienen las respuestas.

1. Resolución Recursiva:

- En la resolución recursiva, el cliente DNS (por ejemplo, tu computadora) realiza una consulta a un servidor DNS y le pide que resuelva un nombre de dominio específico.
- El servidor DNS que recibe la consulta intenta resolver el nombre de dominio consultando a otros servidores DNS si no tiene la información en su caché local.
- Si el servidor DNS puede resolver el nombre de dominio, devolverá la respuesta al cliente DNS que realizó la consulta.
- Si el servidor DNS no puede resolver el nombre de dominio, puede continuar realizando consultas a otros servidores DNS en nombre del cliente hasta que se obtenga una respuesta.

2. Resolución Iterada:

- En la resolución iterada, el servidor DNS que recibe la consulta no intenta resolver directamente el nombre de dominio consultado.
- En cambio, el servidor DNS proporciona al cliente DNS que realizó la consulta una lista de servidores DNS que pueden tener la información necesaria para resolver el nombre de dominio.
- El cliente DNS luego realiza consultas adicionales a estos servidores DNS iterativamente hasta que se obtiene la respuesta deseada.
- El servidor DNS que realiza la consulta iterativa no tiene la responsabilidad de resolver el nombre de dominio por completo; simplemente proporciona información sobre dónde se pueden realizar consultas adicionales.

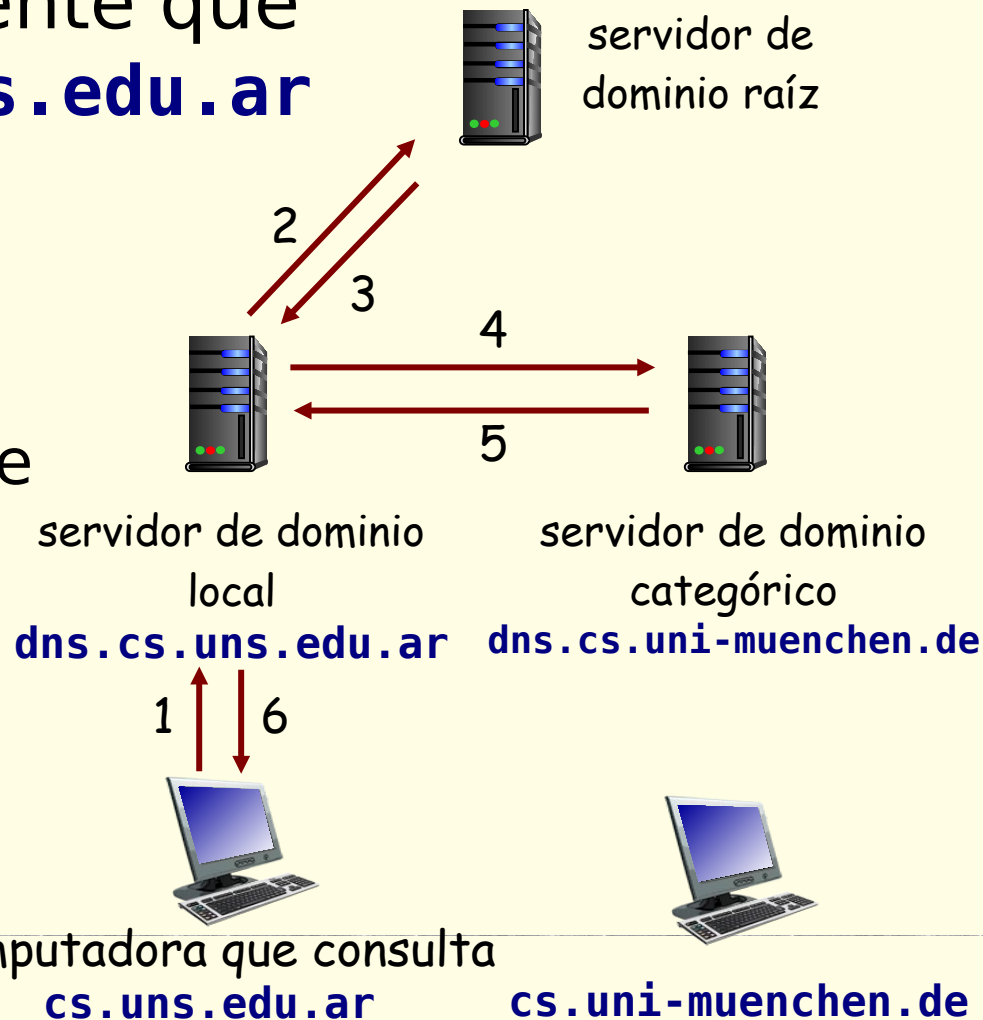
En resumen, la resolución recursiva implica que el servidor DNS realiza consultas en nombre del cliente y devuelve una respuesta completa, mientras que la resolución iterada implica que el servidor DNS proporciona información sobre dónde se pueden realizar consultas adicionales, pero no resuelve el nombre de dominio por completo por sí mismo. La elección entre resolución recursiva e iterativa depende del diseño y la configuración específicos de la red y los servidores DNS involucrados.



Ejemplo de consulta iterada

• Supongamos nuevamente que la computadora **cs.uns.edu.ar** quiere averiguar el **IP** de la computadora **cs.uni-muenchen.de**

→ El servidor raíz no conoce el **IP** buscado, por lo que contesta con la dirección del servidor que posiblemente si conozca ese **IP**



Cache DNS local

- Toda vez que un servidor toma conocimiento de un nuevo mapeo actualiza su **cache individual**
 - ➔ Las entradas de este cache cuentan con un atributo de **tiempo de vida**, es decir, pasado un cierto tiempo desaparecen
- Los mecanismos de actualización de la base de datos de registros y de propagación de estas modificaciones están especificados por un grupo de trabajo de la **IETF**
 - ➔ Este grupo de trabajo produjo el **RFC 2136**



Relación entre nombres e IPs

- El sistema **DNS** tiene que contemplar un conjunto bastante flexible de relaciones entre nombres simbólicos y direcciones **IP**:
 - El caso básico es la relación **uno-a-uno**, por ejemplo **cs.uns.edu.ar** y **200.49.226.11**
 - También existe el caso **muchos-a-uno**, por caso **pop3.cs.uns.edu.ar** y **smtp.cs.uns.edu.ar**
 - Finalmente tenemos el caso **uno-a-muchos**, por caso **google.com** está asociado a un conjunto bastante grande de direcciones **IP**



Registros DNS

- La base de datos distribuida **DNS** se compone de un conjunto de registros **DNS**
- Los registros **DNS** son meras tuplas:
(nombre, valor, tipo, tiempo de vida)
- El rol de cada uno de los campos difiere en función del valor contenido en el campo **tipo**
- Para el **tipo A**, **nombre** contiene el nombre simbólico de una computadora y **valor** su dirección **IP**



Registros DNS

- Para el **tipo NS**, **nombre** contiene un nombre de dominio y **valor** contiene la dirección del servidor categórico para ese dominio
- Para el **tipo CNAME**, **nombre** contiene un alias simbólico que será asociado a un cierto nombre canónico y **valor** contiene ese nombre canónico
- Para el **tipo MX**, **nombre** contiene un dominio y **valor** contiene la dirección del servidor de mail que corresponde a ese dominio



El protocolo DNS

- El protocolo **DNS** usa un **formato de mensaje fijo** para consultas y respuestas
 - ➔ Está especificado formalmente en el **RFC 1034/5**



El protocolo DNS

- Significado de los distintos campos:
 - El **encabezado** tiene un tamaño fijo de de 12 bytes
 - El **cuerpo** es de un tamaño variable (pues depende de la cantidad de respuestas incluidas)
 - El campo **identificación** contiene un valor de 16 bits con el cual se asocian las consultas a las respuestas
 - El campo **flags** permite indicar si se trata de una consulta o una respuesta, si se prefiere o si está disponible una respuesta recursiva, o si la respuesta es categórica



El protocolo DNS

- Significado de los distintos campos (continúa):
 - Cada **consulta** indica el **tipo** de registro buscado así como el valor del campo **nombre** en cuestión
 - Cada **respuesta** contiene el registro **DNS** deseado
 - Cada **respuesta categóricas** contiene el registro **DNS** de un servidor categórico
 - Cada **respuesta adicional** contiene información extra considerada de utilidad



Incorporación de dominios

- Supongamos que queremos arrancar una startup para desarrollar la próxima killer app...
 - Registramos el dominio **killerapp.com** con un gestor (conocidos como **DNS** registrar)
 - El gestor nos brinda el nombre y el **IP** del que será nuestro servidor categórico (primario y secundario)
 - A su vez, inserta dos registros en el server **TLD .com**:
(**killerapp.com, dns1.killerapp.com, NS, 86400**)
(**dns1.killerapp.com, 1.2.3.4, A**)



Vectores de ataque

- Cabe señalar que la jerarquía **DNS** constituye en principio un punto único de fallo de internet
 - ➔ Como tal, ha resultado objeto de diferentes intentos de ataques, con diverso grado de éxito
- Ataques tipo **DDoS** a los servidores raíz:
 - ➔ La idea es tapar con pedidos espurios a uno o más de uno de los servidores raíz
 - ➔ Al día de la fecha nunca funcionó...
 - ➔ ...no obstante, los servidores raíz están **bajo estricta vigilancia armada** las 24 horas del día!



Vectores de ataque

- Ataques tipo **DDoS** a los servidores **TLD**:
 - ➔ ¡Potencialmente más peligroso que el anterior!
- Ataques de redirección:
 - ➔ Variante man-in-the-middle, interceptar y adulterar las consultas y las respuesta a mitad de camino
 - ➔ Variante envenenamiento: contaminar las respuestas almacenadas en el cache de los servidores
- Ataques tipo **DDoS** causados por el **DNS**:
 - ➔ Enviar en cantidad consultas espurias con el **IP** de origen adulterado



Confiabilidad

- La alta replicación de la base de datos **DNS** aporta dos grandes beneficios adicionales:
 - ➔ Mejora notablemente la **confiabilidad**
 - ➔ Permite implementar alguna forma de **balance de carga** a fin de obtener un mejor desempeño
- Se utiliza **UDP** como protocolo de transporte.
 - ➔ De hacer falta integridad, se debe implementar a nivel de la aplicación
 - ➔ ¿Por qué razón no se usa **TCP**?



Herramientas DNS

- El programa **dig** es una herramienta muy práctica para diagnosticar errores en la configuración de los servidores **DNS**
 - ➔ El parámetro **@** permite enviar la consulta a un servidor en particular
 - ➔ El parámetro **+norecurse** permite recorrer a mano la jerarquía de servidores de dominio
 - ➔ El parámetro **+trace** permite inspeccionar las respuestas que recibe el servidor de dominio por defecto al resolver la consulta efectuada



¿Preguntas?

