

*Telefonica*



# WIRESHARK

Aplicaciones



# Aplicaciones

Protocolo DNS



- DNS se creó para proporcionar un método para seguir la pista a nombres y direcciones en Internet, los objetivos son:
  - Crear un espacio de nombres consistente para referirse a los recursos
  - La base de datos y sus actualizaciones, debe mantenerse de forma distribuida, con caches locales
  - Las transacciones entre servidores, deben ser independientes

- Requisitos

- El tamaño de la base de datos, será proporcional al número de hosts que usen el sistema
- Debe adaptarse a los cambios de forma rápida
- Cada organización con responsabilidad de un dominio, deberá tener servidores redundantes
- Los clientes, identificarán un servidor de confianza
- El acceso a la información, es crítico
- Un servidor puede realizar consultas que solo pueden ser respondidas por otro servidor

# Capa de aplicación

## DNS

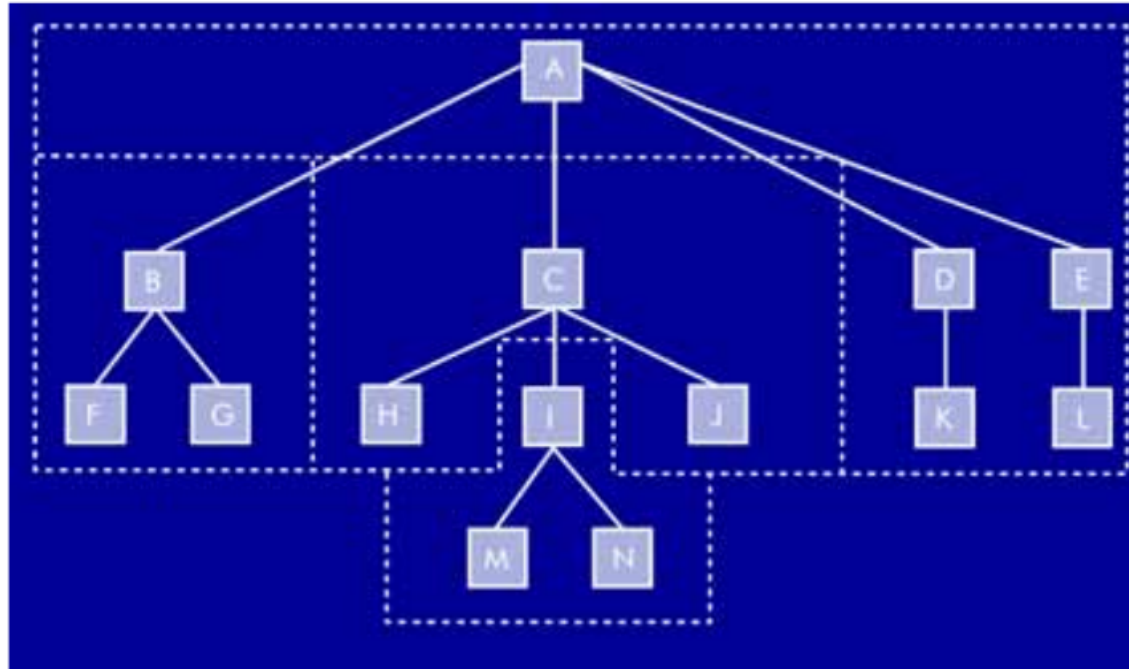
- DNS se estructura en tres componentes principales:
  - Espacio de nombres de dominio y registro de recursos
  - Servidores de nombres, donde se aloja la información de la estructura de un árbol de dominio dentro de unidades llamadas zonas
  - Resolvers o resolutores, son los programas que realizan las preguntas a los servidores. Deben tener acceso, al menos, a un servidor de nombres

- ¿Qué necesita un servidor de nombres para hacer su trabajo?
  - Una lista de servidores raíz donde realizar las consultas externas
  - Una lista de nombres con sus direcciones correspondientes
  - Un servidor secundario

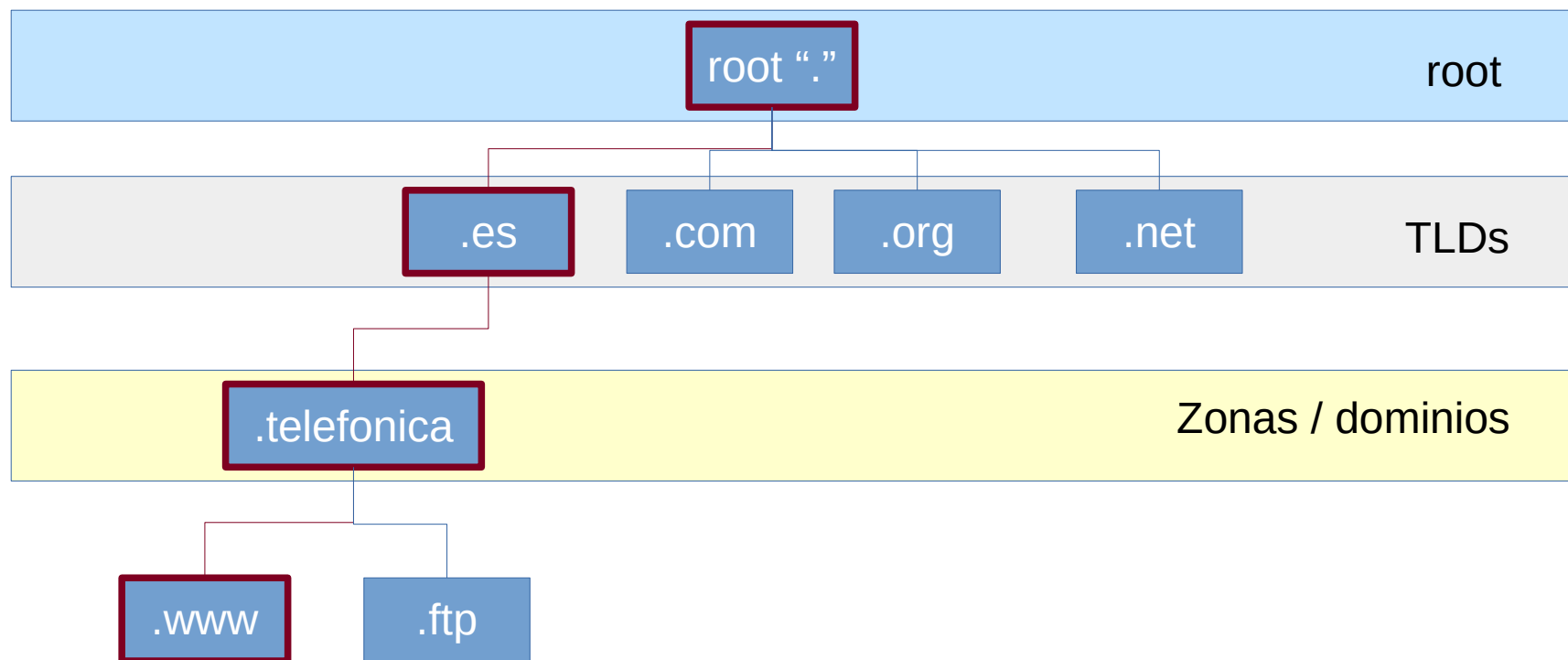
# Capa de aplicación

## DNS

- El espacio de nombres DNS está basado en una estructura jerárquica con un nodo raíz, cada nodo y hoja del árbol, corresponde a un conjunto de recursos



- El nombre de dominio de un nodo, es la secuencia formada por las etiquetas entre ese nodo y el raíz “www.telefonica.es.”

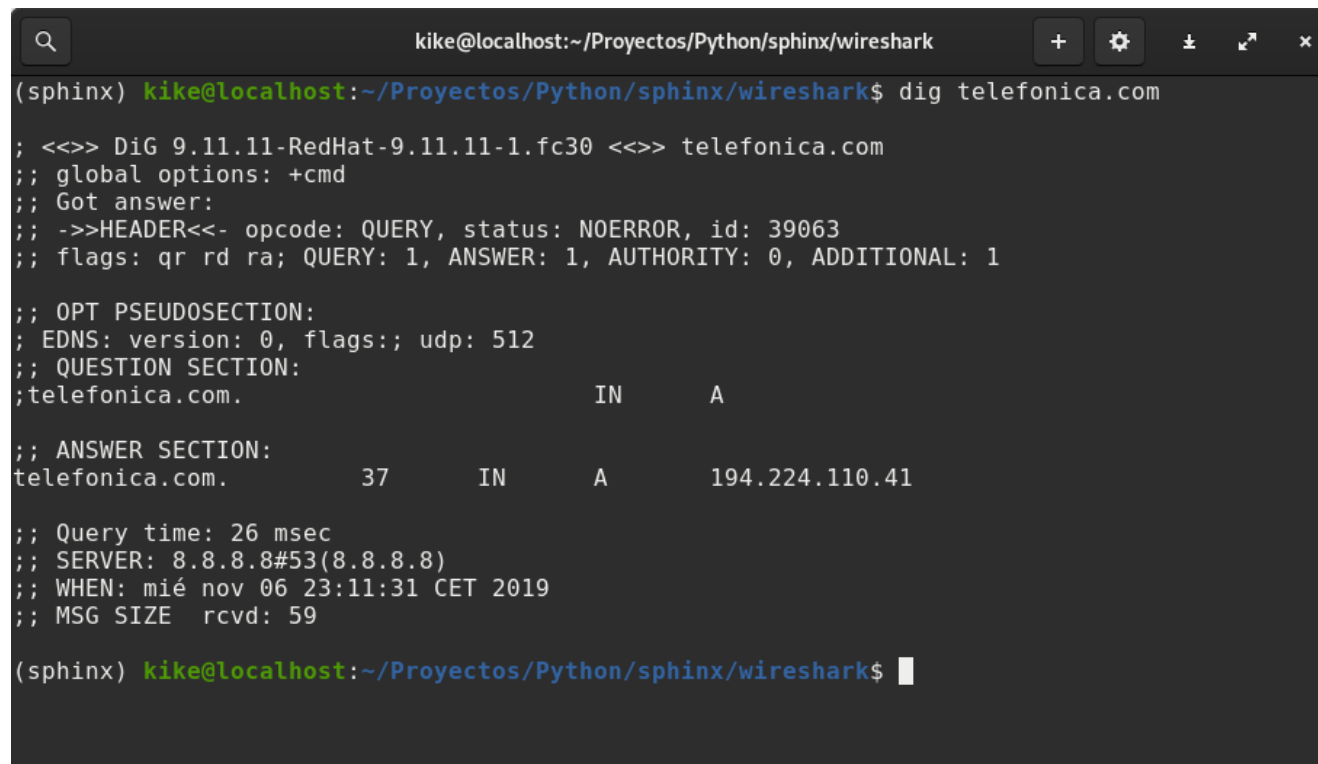




# Capa de aplicación

## DNS

- Los resolutores, reciben peticiones de las aplicaciones de usuario y los traduce a consultas DNS



```
(sphinx) kike@localhost:~/Proyectos/Python/sphinx/wireshark$ dig telefonica.com

; <<>> DiG 9.11.11-RedHat-9.11.11-1.fc30 <<>> telefonica.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 39063
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:;, udp: 512
;; QUESTION SECTION:
;telefonica.com.                IN      A

;; ANSWER SECTION:
telefonica.com.                37      IN      A      194.224.110.41

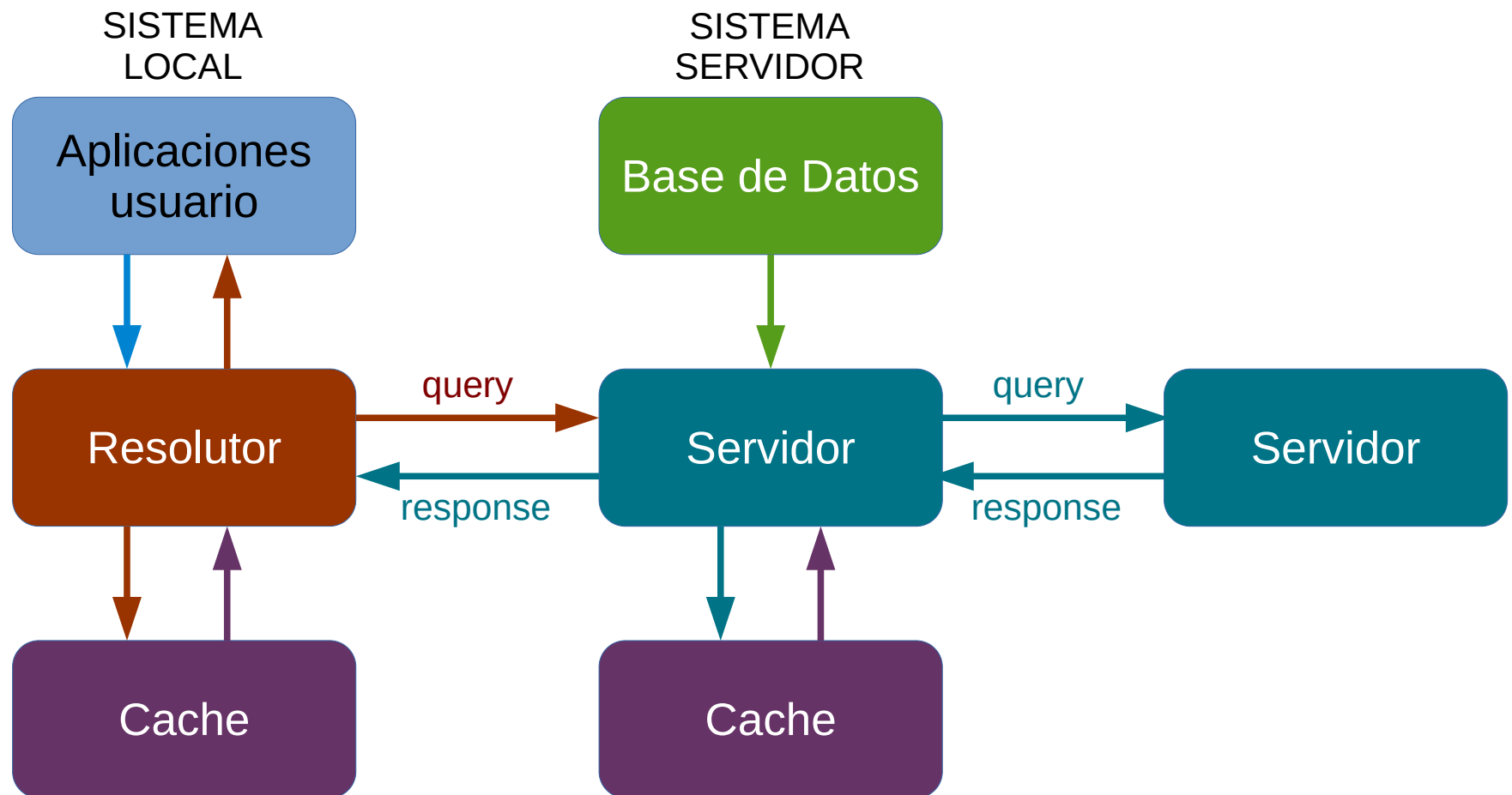
;; Query time: 26 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: mié nov 06 23:11:31 CET 2019
;; MSG SIZE rcvd: 59

(sphinx) kike@localhost:~/Proyectos/Python/sphinx/wireshark$
```

# Capa de aplicación

## DNS

- Funcionamiento del servicio DNS



# Capa de aplicación

## DNS

- Las consultas, son mensajes enviados a un servidor de nombres para obtener una respuesta.
- Se pueden transportar por UDP o TCP, aunque, se suele utilizar
  - TCP para las transferencias de zona
  - UDP para las consultas y respuestas

- Cuando se quiere resolver una consulta, y el servidor local no encuentra lo que busca en su base de datos, puede realizar la búsqueda de dos maneras:
  - Modo recursivo
  - Modo iterativo

- Definimos **zona** como una base de datos completa de un subárbol dentro del espacio de dominio
- Cada zona está bajo una autoridad, y puede delegar la gestión de parte del árbol.
- La información referida a una zona, debe estar almacenada en la base de datos de un servidor, que se dice que tiene autoridad para esa zona.

- Cuando se realizan cambios en la zona del servidor maestro, deben replicarse a todos los servidores secundarios de esa zona, mediante una transferencia de zona
  - Completa, el servidor primario transfiere toda su base de datos al servidor secundario para esa zona
  - Transferencia incremental de zona, solo se transmite la parte modificada de la zona
- La transferencia de zona se realiza de forma automática cuando
  - Se recibe mensaje **NOTIFY** del primario indicando que hay cambios en la zona
  - Ha vencido el tiempo especificado en el campo **REFRESH** del registro autoridad de la zona

- Un nombre de dominio, identifica a un nodo y el conjunto de información de recurso con un nombre, está compuesto por distintos registros de recursos (Rrs)
- Un registro de recurso, tiene:
  - Un propietario, el nombre de dominio
  - Un tipo, que especifica el tipo de recurso
  - Clase IN para Internet
  - TTL, tiempo de vida en la caché

- Tipos de registros de recursos
  - **SOA** Inicio de autoridad, información sobre el nodo superior de una zona
  - **A**, dirección de un nodo
  - **CNAME**, nombre canónico de un alias
  - **HINFO**, información sobre el tipo de nodo
  - **MX**, nombre de un servidor de correo para un dominio
  - **NS**, nombre de un servidor de dominio con autoridad para una zona
  - **PTR**, indica que nombre de host están asignados a cierta dirección IP (resolución inversa)



# Capa de aplicación

## DNS

- El registro SOA, posee los siguientes campos
  - Propietario, nombre de dominio o zona
  - Tipo, SOA
  - Responsable, email del responsable de la zona
  - Número de serie, número de versión de la zona
  - Actualización, tiempo de actualización del secundario
  - Reintentos, tiempo de reenvío de solicitud de transferencia de zona
  - Caducidad, tiempo para descartar su zona como no válida
  - TTL mínimo, cuanto se guardan las respuestas en la caché de usuario

```
kike@kikews:~$ dig +short SOA egalvez.es
ns1024.ui-dns.de. hostmaster.land1.es. 2017060102 28800 7200 604800 300
kike@kikews:~$
```

# Capa de aplicación

## DNS

- El registro A (Address), asigna un nombre de dominio completamente cualificado (FQDN) a una dirección IP

```
kike@kikews:~$ dig +nocmd A egalvez.es
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 24067
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;egalvez.es.                IN      A

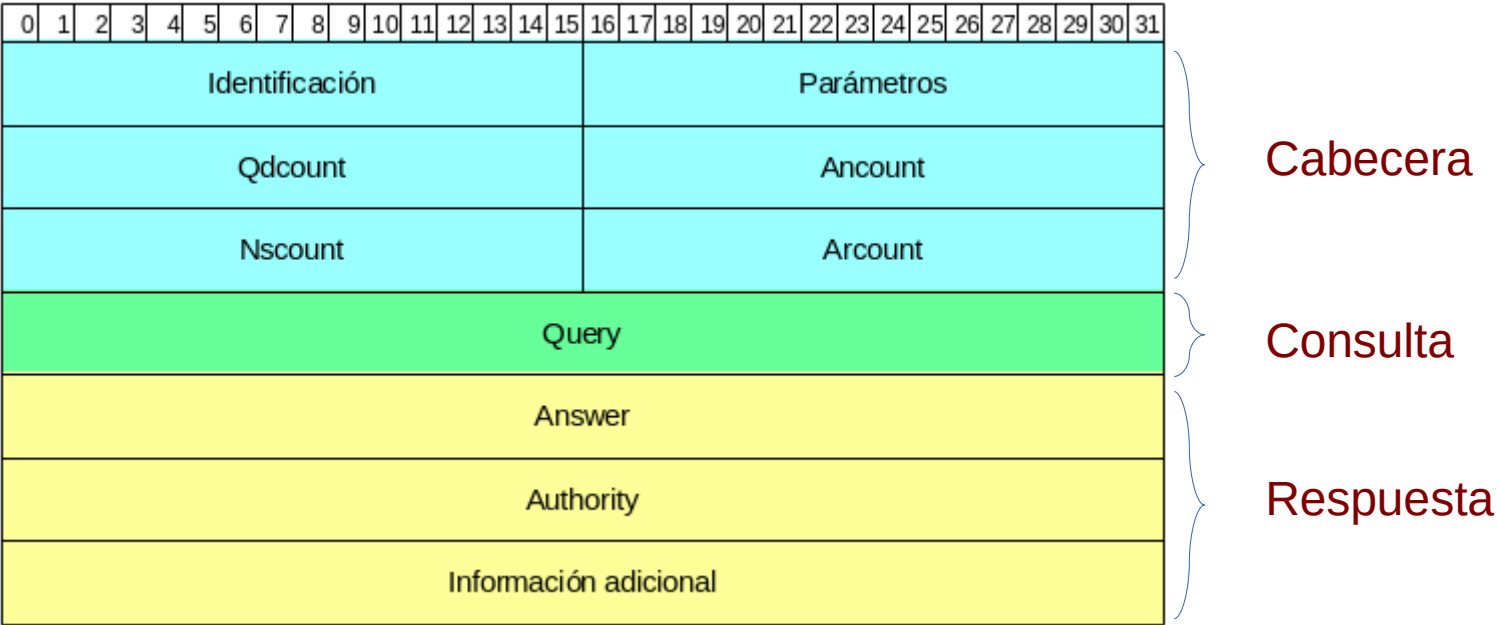
;; ANSWER SECTION:
egalvez.es.                 3600    IN      A      82.223.15.87

;; Query time: 66 msec
;; SERVER: 80.58.61.250#53(80.58.61.250)
;; WHEN: Sun Sep 09 19:51:45 CEST 2018
;; MSG SIZE rcvd: 55
```

# Capa de aplicación

## DNS

- Formato de los mensajes



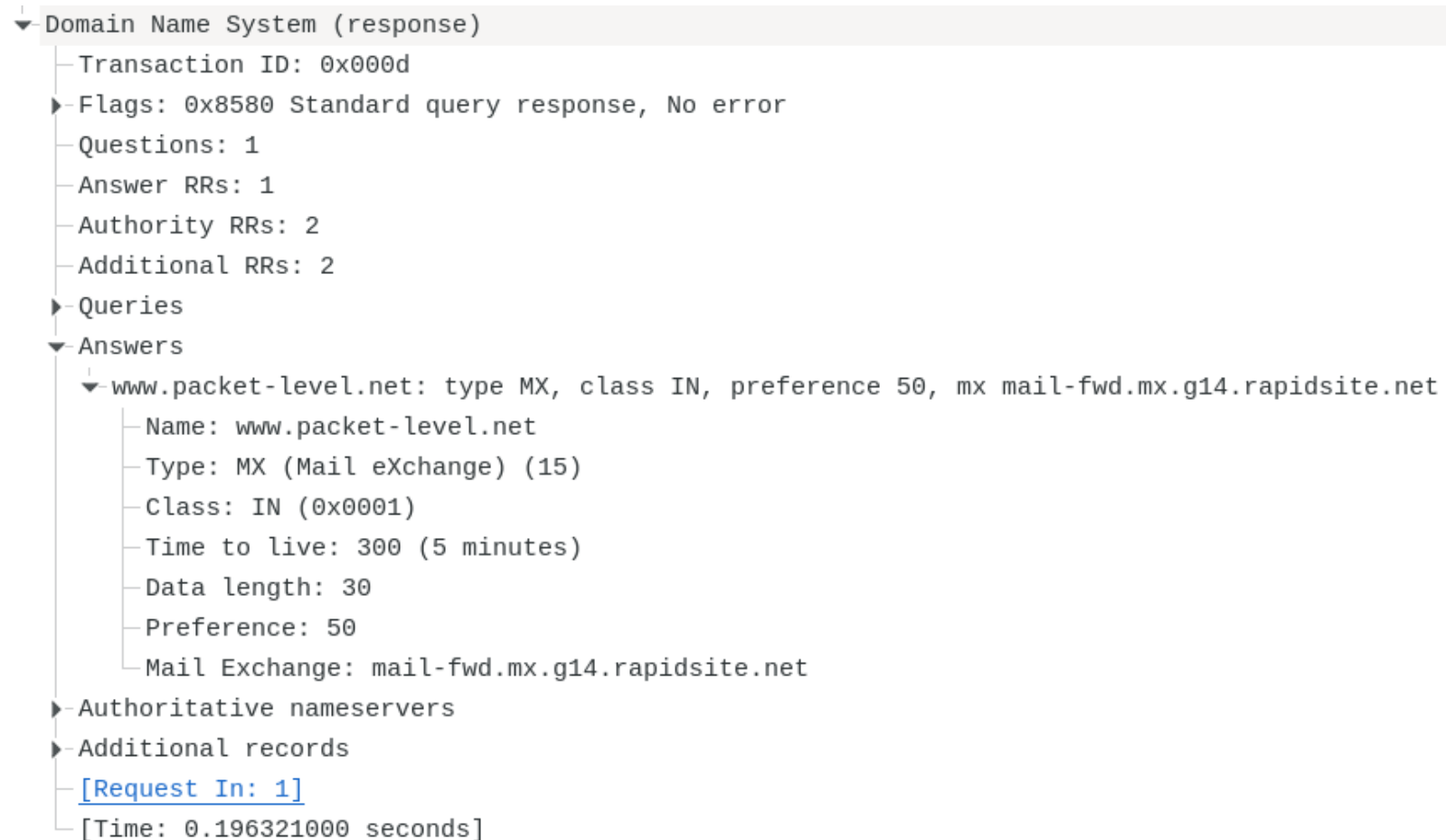
- Formato de una consulta en wireshark

```
▼ Domain Name System (query)
  Transaction ID: 0xa570
  ▼ Flags: 0x0100 Standard query
    0... .. = Response: Message is a query
    .000 0... .. = Opcode: Standard query (0)
    .... 0. .... = Truncated: Message is not truncated
    .... 1 .... = Recursion desired: Do query recursively
    .... 0. .... = Z: reserved (0)
    .... 0 .... = Non-authenticated data: Unacceptable
  Questions: 1
  Answer RRs: 0
  Authority RRs: 0
  Additional RRs: 0
  ▼ Queries
    ▼ www.nmap.org: type A, class IN
      Name: www.nmap.org
      [Name Length: 12]
      [Label Count: 3]
      Type: A (Host Address) (1)
      Class: IN (0x0001)
```

# Capa de aplicación

## DNS

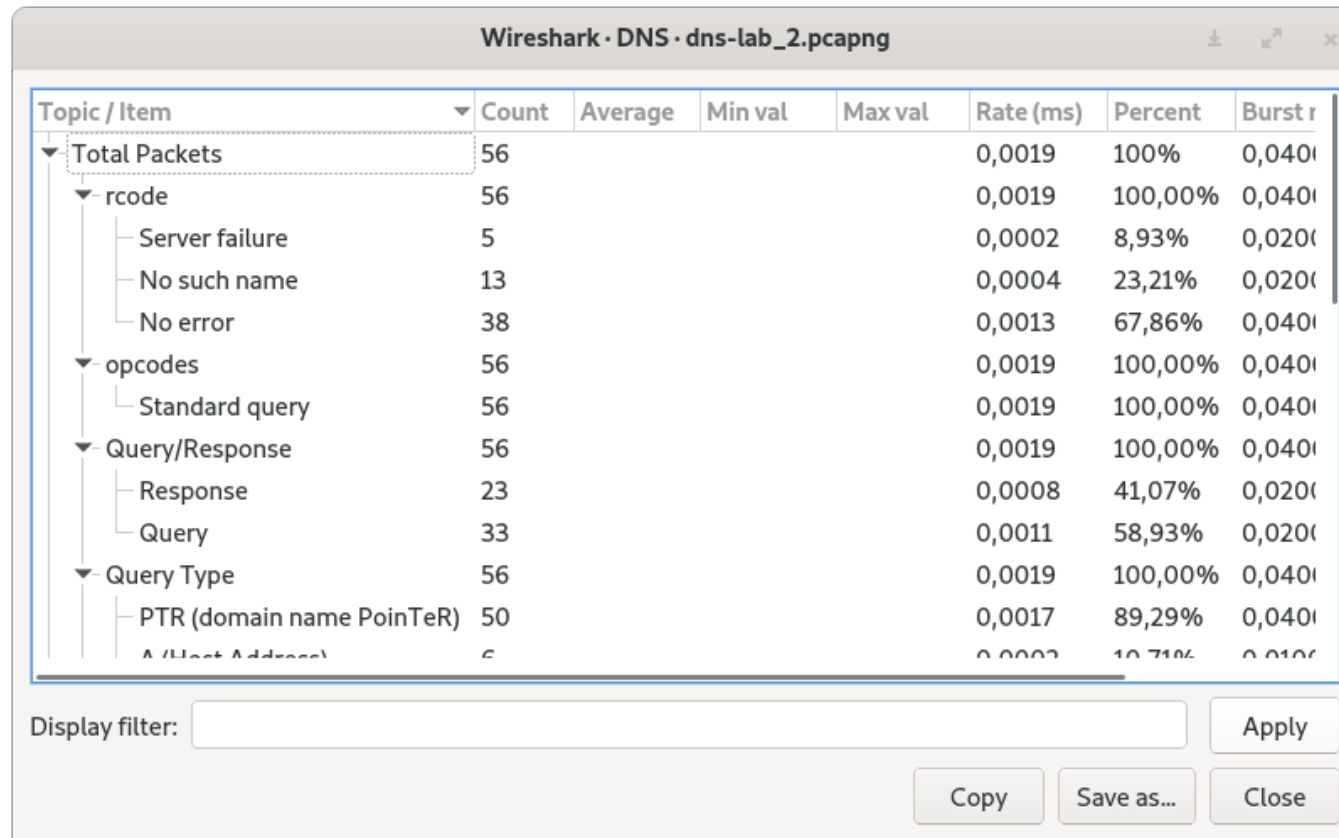
- Formato de una respuesta en wireshark



# Capa de aplicación

## DNS

- Estadísticas DNS, desde el menú **Statistics → DNS**



The image shows the Wireshark Statistics window for the file 'dns-lab\_2.pcapng'. The window displays a hierarchical tree of statistics on the left and a corresponding table of data on the right. The table has columns for Topic / Item, Count, Average, Min val, Max val, Rate (ms), Percent, and Burst r. The data is organized into several categories: Total Packets, rcode, opcodes, Query/Response, and Query Type. The 'Total Packets' category shows a count of 56. The 'rcode' category shows counts for Server failure (5), No such name (13), and No error (38). The 'opcodes' category shows a count of 56 for Standard query. The 'Query/Response' category shows counts for Response (23) and Query (33). The 'Query Type' category shows counts for PTR (domain name PoinTeR) (50) and A (Host Address) (6).

Topic / Item	Count	Average	Min val	Max val	Rate (ms)	Percent	Burst r
▼ Total Packets	56				0,0019	100%	0,0400
▼ rcode	56				0,0019	100,00%	0,0400
Server failure	5				0,0002	8,93%	0,0200
No such name	13				0,0004	23,21%	0,0200
No error	38				0,0013	67,86%	0,0400
▼ opcodes	56				0,0019	100,00%	0,0400
Standard query	56				0,0019	100,00%	0,0400
▼ Query/Response	56				0,0019	100,00%	0,0400
Response	23				0,0008	41,07%	0,0200
Query	33				0,0011	58,93%	0,0200
▼ Query Type	56				0,0019	100,00%	0,0400
PTR (domain name PoinTeR)	50				0,0017	89,29%	0,0400
A (Host Address)	6				0,0002	10,71%	0,0100

Display filter:  Apply

Copy Save as... Close

# Capa de aplicación

## DNS errores en el servicio

- Nos podemos encontrar varios tipos de problemas en el servicio DNS
  - No se puede resolver el nombre de dominio
  - Servicio DNS funciona muy lento

- Vulnerabilidades y debilidades del protocolo DNS
  - Suplantación de servidor, que nos permite
    - Envenenamiento de la caché
    - Se puede realizar un ataque de denegación de servicio por amplificación utilizando un servidor DNS
  - Obtención de base de datos mediante transferencia de zona



- Filtros típicos para análisis de tráfico DNS
  - Filtrando por query

```
dns.flags.response == 0
```

- Filtrando por responses

```
dns.flags.response == 1
```

- Si no hay ningún error en la consulta

```
dns.flags.rcode == 0
```

- Filtros típicos para análisis de tráfico DNS
  - Si estamos buscando una URL específica

```
dns.qry.name == "URL"  
dns.qry.name contains "parte URL"
```

- Filtrando por query inversa

```
dns.qry.type == "PTR"
```

- Filtrar por un aviso de cambio de zona

```
dns.flags.opcode == 4
```

# Capa de aplicación

## DNS filtrado

- Filtros típicos para análisis de tráfico DNS
  - Filtrar por la longitud de la query

```
dns.qry.name.len > 25
```

- Filtrando por la longitud de response

```
dns.resp.len > 4
```

# Aplicaciones

## Laboratorio 1 - Protocolo DNS

Analizar los problemas **DNS** que se ven en los siguientes tres archivos

# Aplicaciones

Protocolo HTTP



- El propósito del protocolo HTTP es permitir la transferencia de archivos (principalmente HTML) entre un navegador y un servidor web
- Se basa en URL (Localizador Uniforme de Recursos) para indicar el recurso que hace referencia una petición HTTP.

- Se basa en un paradigma de peticiones y respuestas
- El cliente se conecta al servidor y envía una petición, donde indica el método, el recurso y la versión

```
GET /home.html HTTP/1.1  
Accept: text/html
```

- El servidor responde con un código de éxito o error

```
HTTP/1.1 200 OK
```

- La sintaxis de la petición es:

```
http://direccion:puerto/nombre_del_recurso
```

- Donde dirección es el nombre de un dominio o una dirección IP
- Puerto donde se envía la petición, en caso de no indicarse, será el puerto 80
- Nombre del recurso al que se quiere acceder



- Una petición, ha de incluir el método que se aplica al recurso. Los métodos más usados son:
  - **GET**, pide una representación del recurso especificado
  - **HEAD**, igual a GET, salvo que el servidor no tiene que devolver contenido, solo las cabeceras
  - **POST**, envían datos que se incluirán en el cuerpo de la petición
  - **PUT**, sube o carga un recurso
  - **DELETE**, borra el recurso especificado
  - **OPTIONS**, devuelve los métodos HTTP soportados por el servidor

- Los recursos HTTP se solicitan mediante métodos que tienen la siguiente estructura

```
<VERBO> <RECURSO> HTTP/ <VERSION> <CRLF>  
<CABECERAS> <CRLF>  
<CRLF>
```

```
GET / HTTP/1.1\r\n
```

- La petición, siempre termina en blanco, se envía <CRLF> para que el servidor sepa que ya hemos terminado

- Las cabeceras, se aplican, tanto en las peticiones, como en las respuestas. Pueden ser:
  - Cabeceras generales
  - De petición, permiten al cliente, pasar información adicional al servidor
  - De respuesta, pasan información del servidor al cliente sobre la respuesta y el recurso solicitado
  - De entidad, aportan información sobre el contenido del mensaje

- Dentro de las cabeceras de petición, podemos encontrar
  - Accept, indican el tipo de respuesta que aceptan
  - Accept-Encoding, el tipo de codificación
  - Host, especifica la maquina y el puerto del recurso pedido
  - User-Agent, información sobre el agente que genera la petición
  - Proxy-Authorization, para identificación del cliente

- Una petición a un recurso

```
▼ Hypertext Transfer Protocol
  ▶ GET /whatsup.html HTTP/1.1\r\n
  — Host: www.chappellu.com\r\n
  — User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:16.0) Gecko/20100101 Firefox/16.0\r\n
  — Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
  — Accept-Language: en-US,en;q=0.5\r\n
  — Accept-Encoding: gzip, deflate\r\n
  — Connection: keep-alive\r\n
  — \r\n
  — \[Full request URI: http://www.chappellu.com/whatsup.html\]
  — [HTTP request 1/2]
  — \[Response in frame: 9\]
  — \[Next request in frame: 10\]
```

- Códigos de estado
  - 1xx → Informativo, no se usa
  - 2xx → Correcto, la acción se ha aceptado correctamente
  - 3xx → Redirección, se debe realizar alguna acción adicional para completar la petición.
  - 4xx → Error de cliente, la petición no se puede conceder.
  - 5xx → Error de servidor, la petición es correcta

- Dentro de las cabeceras de respuesta, podemos encontrar
  - Age, estimación del tiempo transcurrido desde que se creó la respuesta
  - Proxy-Authenticate, indica el esquema de autenticación
  - Server, información sobre el servidor
  - Public, lista de métodos soportados por el servidor
  - Warning, usada para aportar información adicional sobre el estado de la respuesta

- Respuesta HTTP

```
▼ Hypertext Transfer Protocol
  ▶ HTTP/1.1 200 OK\r\n
  - Cache-Control: no-cache\r\n
  - Date: Sat, 07 Jan 2012 21:54:16 GMT\r\n
  - Pragma: no-cache\r\n
  - Content-Type: text/html; charset=iso-8859-1\r\n
  - Last-Modified: Sat, 07 Jan 2012 21:54:14 GMT\r\n
  - Accept-Ranges: bytes\r\n
  - Server: Microsoft-IIS/6.0\r\n
  - P3P: CP="CA0 DSP COR CURa ADMa DEVa TAIa PSAa PSDa IVAi IVDi CONi OUR SAMo OTRo BUS PHY ONL UNI PUR COM NAV INT DEM CNT STA PRE"\r\n
  - From: N710\r\n
  - Set-Cookie: SWID=A04FAD47-4E17-4FDA-8F47-4BF82CB53BDD; path=/; expires=Sat, 07-Jan-2032 21:54:16 GMT; domain=.go.com;\r\n
  - Cache-Expires: Sat, 07 Jan 2012 21:54:29 GMT\r\n
  ▶ Content-Length: 72010\r\n
  - Vary: Accept-Encoding\r\n
  - Content-Encoding: gzip\r\n
  - Connection: Keep-Alive\r\n
  - \r\n
```



- El protocolo HTTPS es una versión segura del protocolo HTTP que implementa un canal seguro basado en SSL entre el navegador y el servidor.
- El puerto sobre el que trabaja es el 443

- Filtros típicos para análisis HTTP
  - Envío de peticiones HTTP al servidor

```
http.request.method == "GET"
```

- Código de respuesta por parte del servidor a una petición

```
http.response.code == "404"
```

- Petición de un recurso por parte del cliente

```
http.request.uri contains "metasploit"
```

- Filtros típicos para análisis HTTP
  - A que tipo de servidor estoy conectado

```
http.server == "nginx"
```

- Peticiones realizadas desde un navegador concreto

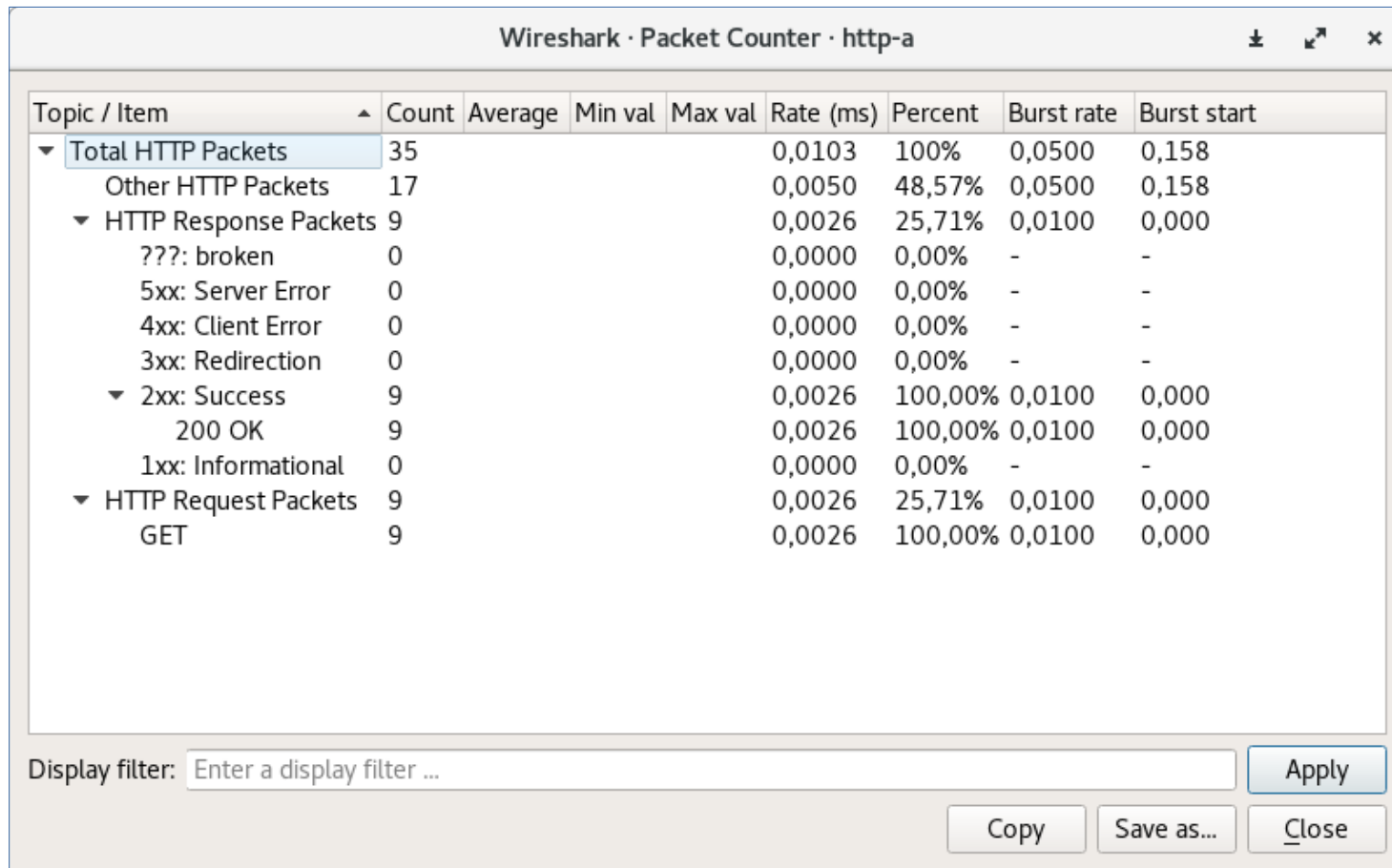
```
http.user_agent matches "[M|m]ozilla"
```

- Información del host a la que se hace la petición

```
http.host contains "hacking"
```

- Para obtener una copia de los recursos descargados en la traza HTTP
  - Habilitar **Allow subdissector to reassemble TCP streams**
  - Desde **File → Export Objects → HTTP**, seleccionamos el recurso a descargar y pulsamos “Guardar como”
  - Recordar desmarcar el reensamblaje de segmentos TCP

- Contador de paquetes http
  - Statistics → HTTP → Packet Counter

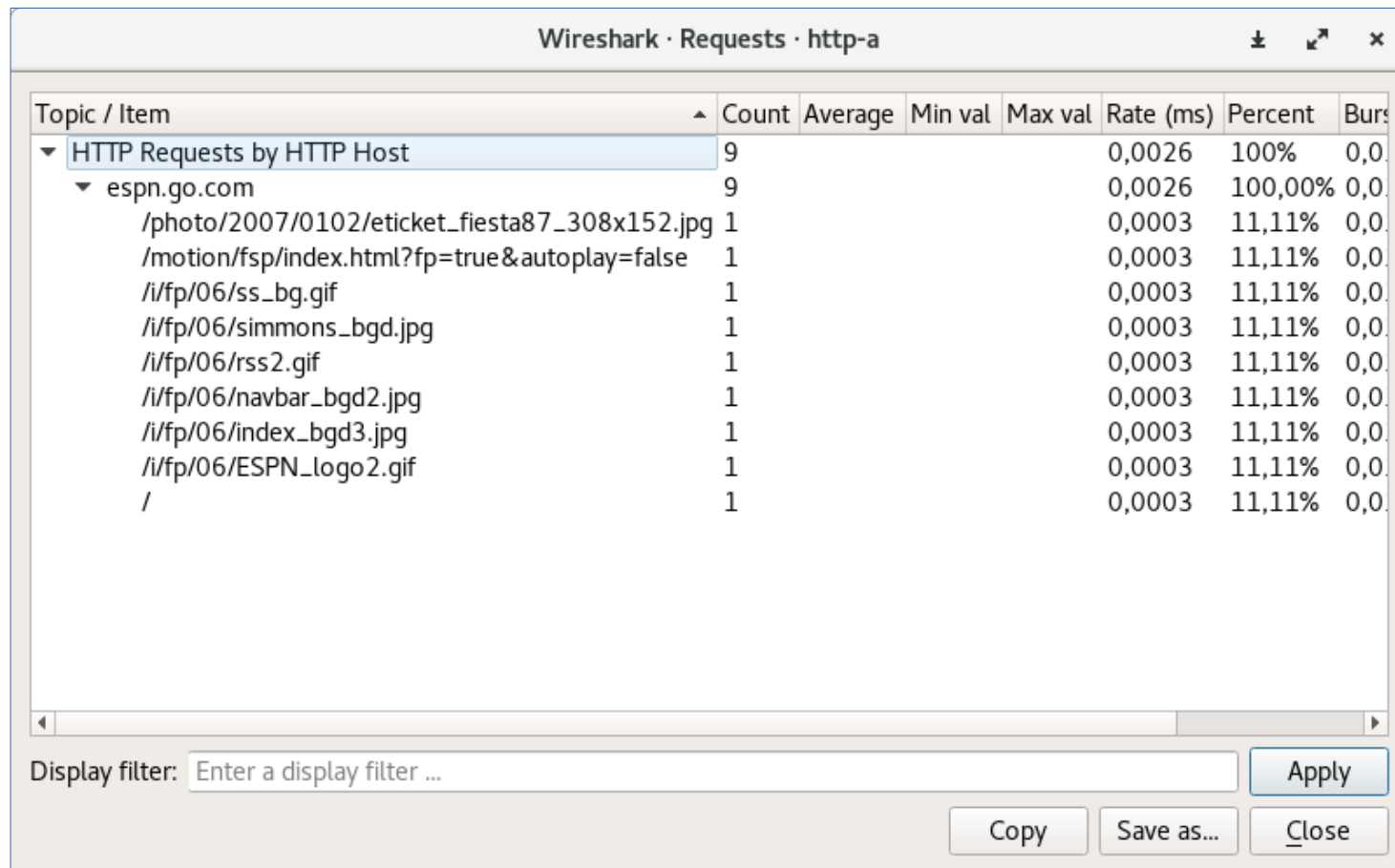


The image shows the 'Wireshark · Packet Counter · http-a' window. It displays a table of HTTP statistics. The table has columns for Topic / Item, Count, Average, Min val, Max val, Rate (ms), Percent, Burst rate, and Burst start. The data is organized into a tree structure under 'Total HTTP Packets'.

Topic / Item	Count	Average	Min val	Max val	Rate (ms)	Percent	Burst rate	Burst start
▼ Total HTTP Packets	35				0,0103	100%	0,0500	0,158
Other HTTP Packets	17				0,0050	48,57%	0,0500	0,158
▼ HTTP Response Packets	9				0,0026	25,71%	0,0100	0,000
??? : broken	0				0,0000	0,00%	-	-
5xx: Server Error	0				0,0000	0,00%	-	-
4xx: Client Error	0				0,0000	0,00%	-	-
3xx: Redirection	0				0,0000	0,00%	-	-
▼ 2xx: Success	9				0,0026	100,00%	0,0100	0,000
200 OK	9				0,0026	100,00%	0,0100	0,000
1xx: Informational	0				0,0000	0,00%	-	-
▼ HTTP Request Packets	9				0,0026	25,71%	0,0100	0,000
GET	9				0,0026	100,00%	0,0100	0,000

At the bottom of the window, there is a 'Display filter:' input field with the placeholder text 'Enter a display filter ...'. To the right of this field are three buttons: 'Apply', 'Copy', and 'Save as...'. At the bottom right corner, there is a 'Close' button.

- Contador de paquetes http
  - Statistics → HTTP → Request



The screenshot shows the 'Wireshark · Requests · http-a' window. The 'Statistics' pane is expanded to 'HTTP Requests by HTTP Host'. The table below represents the data shown in the window.

Topic / Item	Count	Average	Min val	Max val	Rate (ms)	Percent	Burst
HTTP Requests by HTTP Host	9				0,0026	100%	0,0
espn.go.com	9				0,0026	100,00%	0,0
/photo/2007/0102/eticket_fiesta87_308x152.jpg	1				0,0003	11,11%	0,0
/motion/fsp/index.html?fp=true&autoplay=false	1				0,0003	11,11%	0,0
/i/fp/06/ss_bg.gif	1				0,0003	11,11%	0,0
/i/fp/06/simmons_bgd.jpg	1				0,0003	11,11%	0,0
/i/fp/06/rss2.gif	1				0,0003	11,11%	0,0
/i/fp/06/navbar_bgd2.jpg	1				0,0003	11,11%	0,0
/i/fp/06/index_bgd3.jpg	1				0,0003	11,11%	0,0
/i/fp/06/ESPN_logo2.gif	1				0,0003	11,11%	0,0
/	1				0,0003	11,11%	0,0

At the bottom of the window, there is a 'Display filter' field with the text 'Enter a display filter ...' and an 'Apply' button. Below this are three buttons: 'Copy', 'Save as...', and 'Close'.

# Aplicaciones

## Laboratorio 2 - Protocolo HTTP

Archivo: **Fallo HTTP**, Observar el problema de esta traza  
Qué se descarga en el segundo archivo, ¿se puede obtener una copia de la descarga?

# Aplicaciones

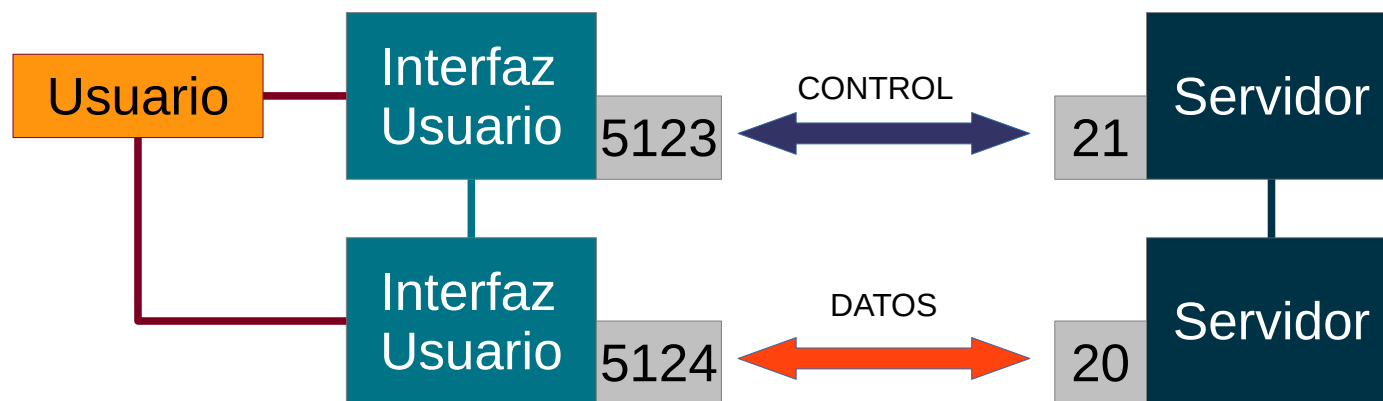
Protocolo FTP





- FTP es un protocolo para la transferencia de archivos en una red cuyo protocolo de transporte es TCP.
- Permite copiar archivos de un sistema a otro, ver listado de directorios y realizar tareas sobre archivos
  - Borrar archivos
  - Renombrar archivos
- FTP confía en TCP para garantizar la integridad de los datos.

- El software de cliente local entabla una conversación con el servidor a través de una conexión de control
- Si se pide una transferencia, se abre una conexión de datos independiente y el archivo se copia a través de ella.



- La comunicación entre el usuario y el servidor, se lleva a cabo, mediante un diálogo.
- El usuario envía una orden y el servidor, devuelve una respuesta

- Login de usuario

File Transfer Protocol (FTP)  
 USER anonymous\r\n  
 Request command: USER  
 Request arg: anonymous  
 [Current working directory: ]



File Transfer Protocol (FTP)  
 331 Please specify the password.\r\n  
 Response code: User name okay, need password (331)  
 Response arg: Please specify the password.  
 [Current working directory: ]



File Transfer Protocol (FTP)  
 PASS anypwd\r\n  
 Request command: PASS  
 Request arg: anypwd  
 [Current working directory: ]



- Para gestionar la comunicación, se dispone de un grupo de comandos de control
  - Autenticación, para iniciar la conexión y hacer login en el sistema
  - Transferencia de archivos, subir y bajar archivos desde el servidor
  - Gestión de archivos, cambiar el nombre del archivo o borrarlo, cambiar de directorio, etc...
  - Control, indicar si el archivo se va a transferir en modo ASCII o Binario

- Para gestionar la comunicación, se dispone de un grupo de comandos de control

```
C:\Users\kike>ftp
```

```
ftp> help
```

```
Los comandos se pueden abreviar. Comandos:
```

!	delete	literal	prompt	send
?	debug	ls	put	status
append	dir	mdelete	pwd	trace
ascii	disconnect	mdir	quit	type
bell	get	mget	quote	user
binary	glob	mkdir	recv	verbose
bye	hash	mls	remotehelp	
cd	help	mput	rename	
close	lcd	open	rmdir	

```
ftp> quit
```

- Códigos de respuesta, el primer dígito
  - 1yz, respuesta preliminar positiva
  - 2yz, respuesta de finalización positiva, la acción requerida se ha terminado satisfactoriamente
  - 3yz, respuesta intermedia positiva, la orden se ha aceptado, pero se espera más información para completarla
  - 4yz, respuesta de finalización negativa, no se ha aceptado la orden y la acción no se ha realizado
  - 5yz, respuesta de finalización negativa permanente, el usuario, no debería repetir la misma petición

- Códigos de respuesta, segundo dígito
  - x1z, información, respuestas a solicitudes de información
  - x2z, conexiones, respuestas referidas a las conexiones de control
  - x3z, Autenticación y cuenta, respuestas para el proceso de entrada al sistema
  - x4z, sin especificar
  - x5z, sistema de ficheros, indican el estado del sistema de ficheros



- Ejemplos
  - 200, orden correcta
  - 500, error de sintaxis, comando desconocido
  - 421, servicio no disponible, cerrando la conexión de control
  - 425, no se puede abrir la conexión de datos
  - 230, usuario conectado
  - 331, usuario OK, necesita contraseña

- Para solventar errores de conexión en transferencias de archivos de gran tamaño, se implementa la función reinicio.
- El FTP emisor transmite bloques que contienen marcadores en puntos adecuados, cada vez que el receptor recibe un marcador, guarda los datos a disco y toma nota de la posición del marcador.
- En caso de fallo de sistema, el usuario puede invocar un comando de reinicio pasando el marcador como argumento, el sistema ejecuta la transferencia a partir de ese marcador.

- Filtros típicos para análisis de tráfico FTP
  - Login con usuario y password

```
ftp.request.command=="USER" || ftp.request.command=="PASS"
```

- Filtrando por argumento del usuario

```
ftp.request.arg=="anonymous"
```

- Respuesta del servidor cuando el login es correcto

```
ftp.response.code == 230
```

- Filtros típicos para análisis de tráfico FTP
  - Crear un directorio llamado backup

```
ftp.request.command=="MKD" && ftp.request.arg=="backup"
```

- Respuesta de que el directorio se ha creado correctamente

```
ftp.response.code == 257
```

# Aplicaciones

## Laboratorio 3 - Protocolo FTP

Analizar el contenido del archivo y ver si es posible obtener una copia de lo descargado  
Que le pasa a esta sesión FTP

# Aplicaciones

## Laboratorio 4

Crear un perfil para la detección de problemas y errores para TCP, HTTP, DNS y FTP, se debe incluir

- Reglas de coloreado
- Detección de códigos de error
- Detalles particulares de cada protocolo.

# FIN

