Telefónica

WIRESHARK

Aplicaciones

Aplicaciones

Protocolo DNS

Capa de aplicación

DNS

- DNS se creo para proporcionar un método para seguir la pista a nombres y direcciones en Internet
  - Es una base de datos distribuida, organizada de forma jerárquica formando un sistema de dominios.
  - Fundamentalmente, se encarga de traducir direcciones IP de recursos de red a nombres fácilmente legibles

Capa de aplicación

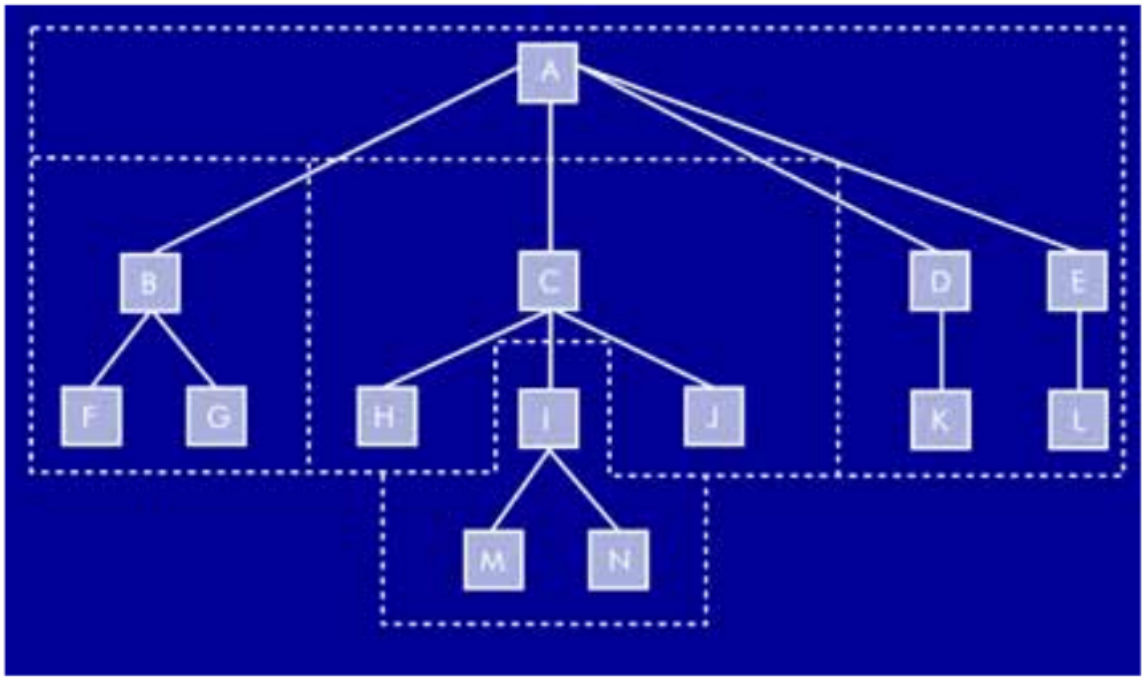
DNS

- DNS se estructura en tres componentes principales:
  - Espacio de nombres
  - Servidores de nombres
  - Resolvers o resolutores

Capa de aplicación

DNS

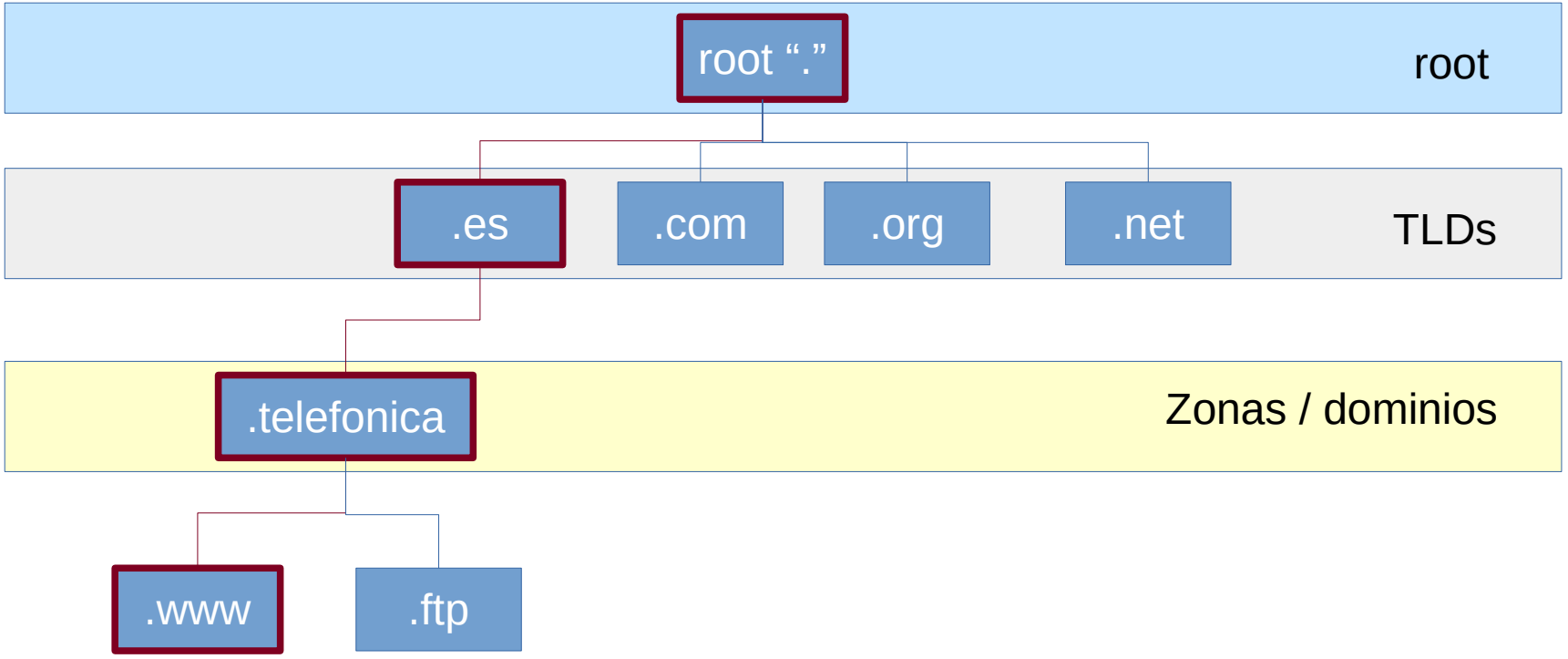
- El espacio de nombres DNS está basado en una estructura jerárquica con un nodo raíz, nodos de primer nivel (TLD) y nodos de segundo nivel. La jerarquía continúa hasta un nodo final que representa un recurso



Capa de aplicación

DNS

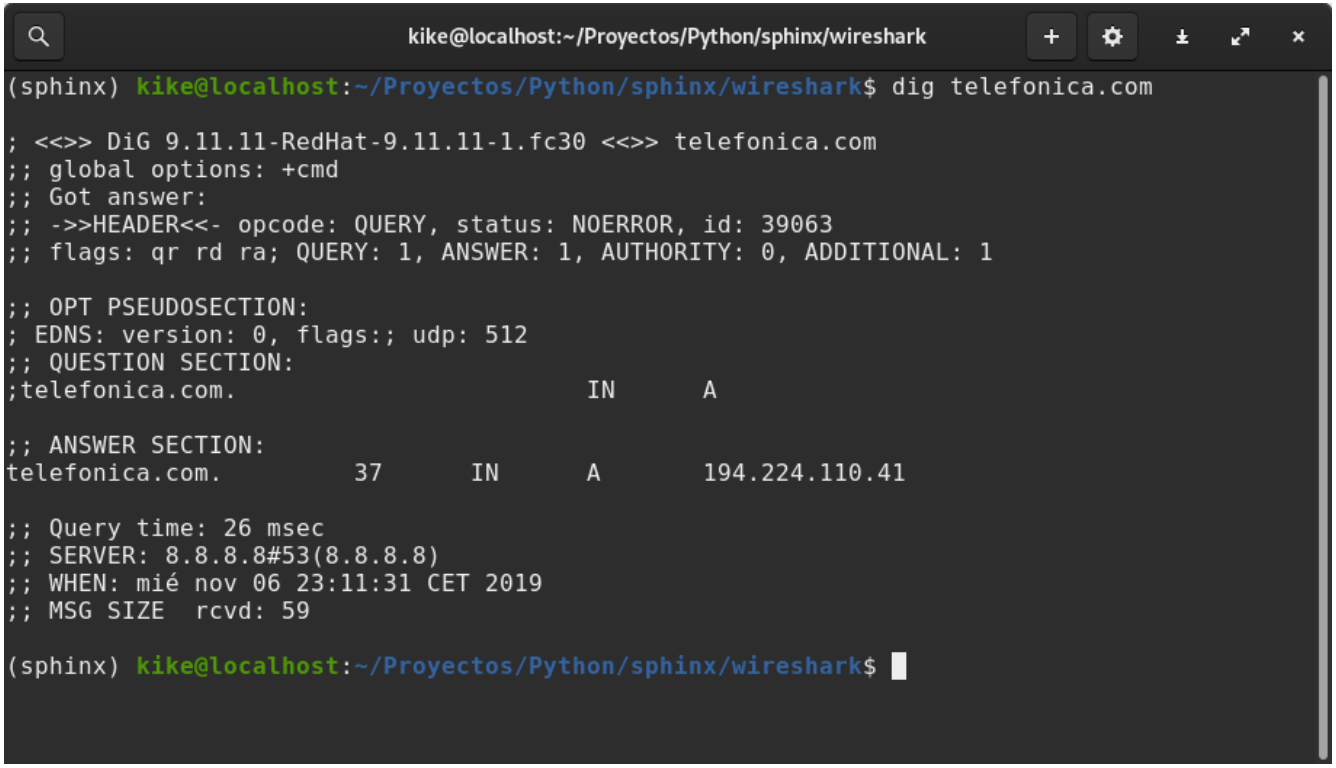
- El nombre de dominio de un nodo, es la secuencia formada por las etiquetas entre ese nodo y el raíz “www.telefonica.es.”



Capa de aplicación

DNS

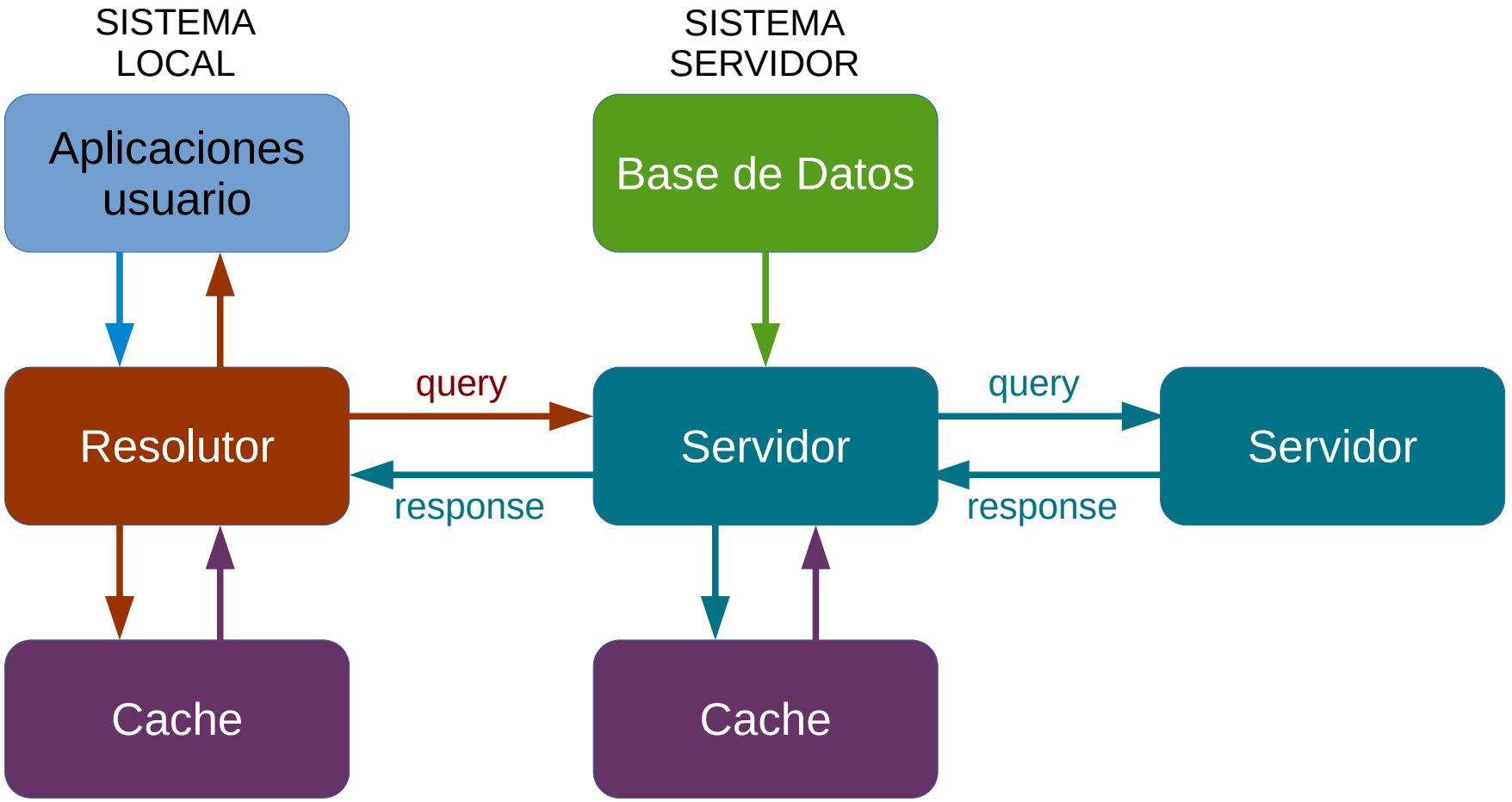
- Los resolutores, reciben peticiones de las aplicaciones de usuario y los traduce a consultas DNS



Capa de aplicación

DNS

- Funcionamiento del servicio DNS



Capa de aplicación

DNS

- 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

Capa de aplicación

DNS

- 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.

Capa de aplicación

DNS

- ¿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

- 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 e 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 SOA de la zona

Capa de aplicación

DNS

- Para acceder a DNS, se utiliza tanto TCP como UDP
  - TCP para las transferencias de zona
  - UDP para las consultas y respuestas

Capa de aplicación

DNS

- La base de datos de DNS, se almacena como una serie de entradas de texto, que se denomina "**Registro de Recurso**" (RR), hay varios tipos de recurso
  - 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.1and1.es. 2017060102 28800 7200 604800 300
kike@kikews:~$
```

Capa de aplicación

DNS

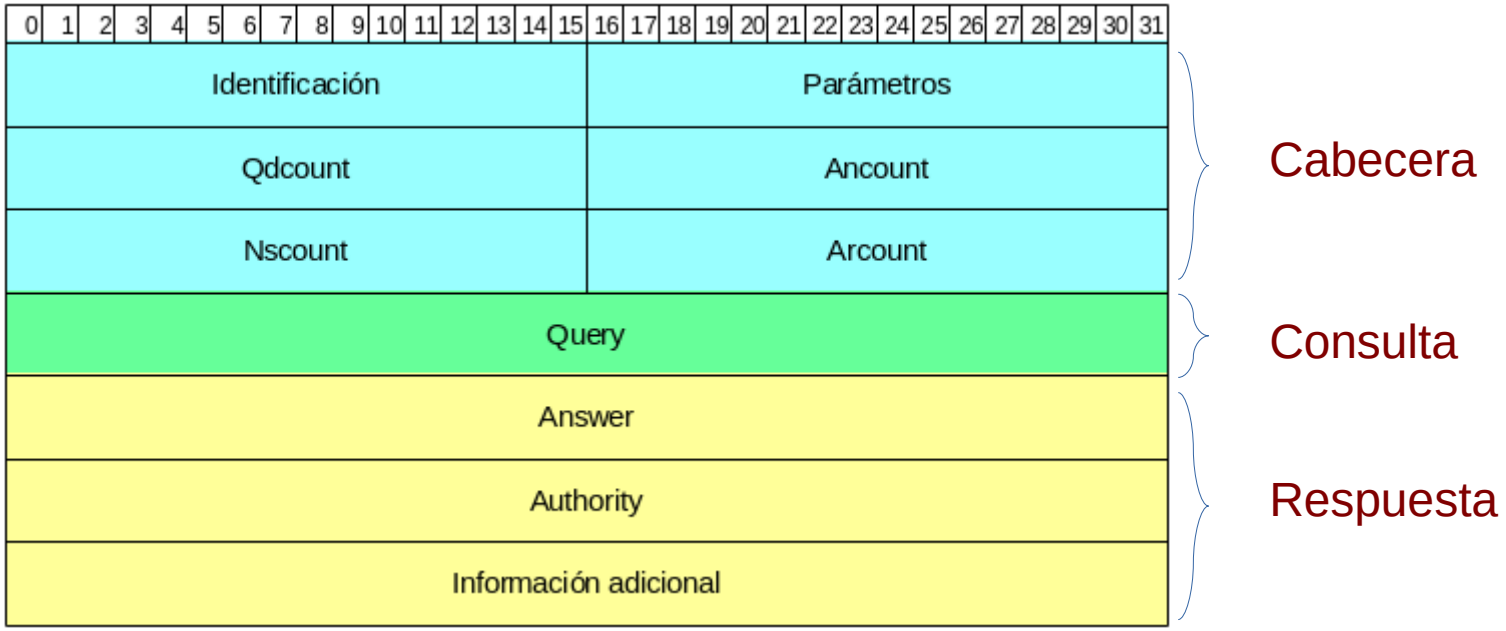
- El registro A (Address), asigna un nombre de dominio completamente cualificado (FQDN) a una dirección IP

```
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



Capa de aplicación

DNS

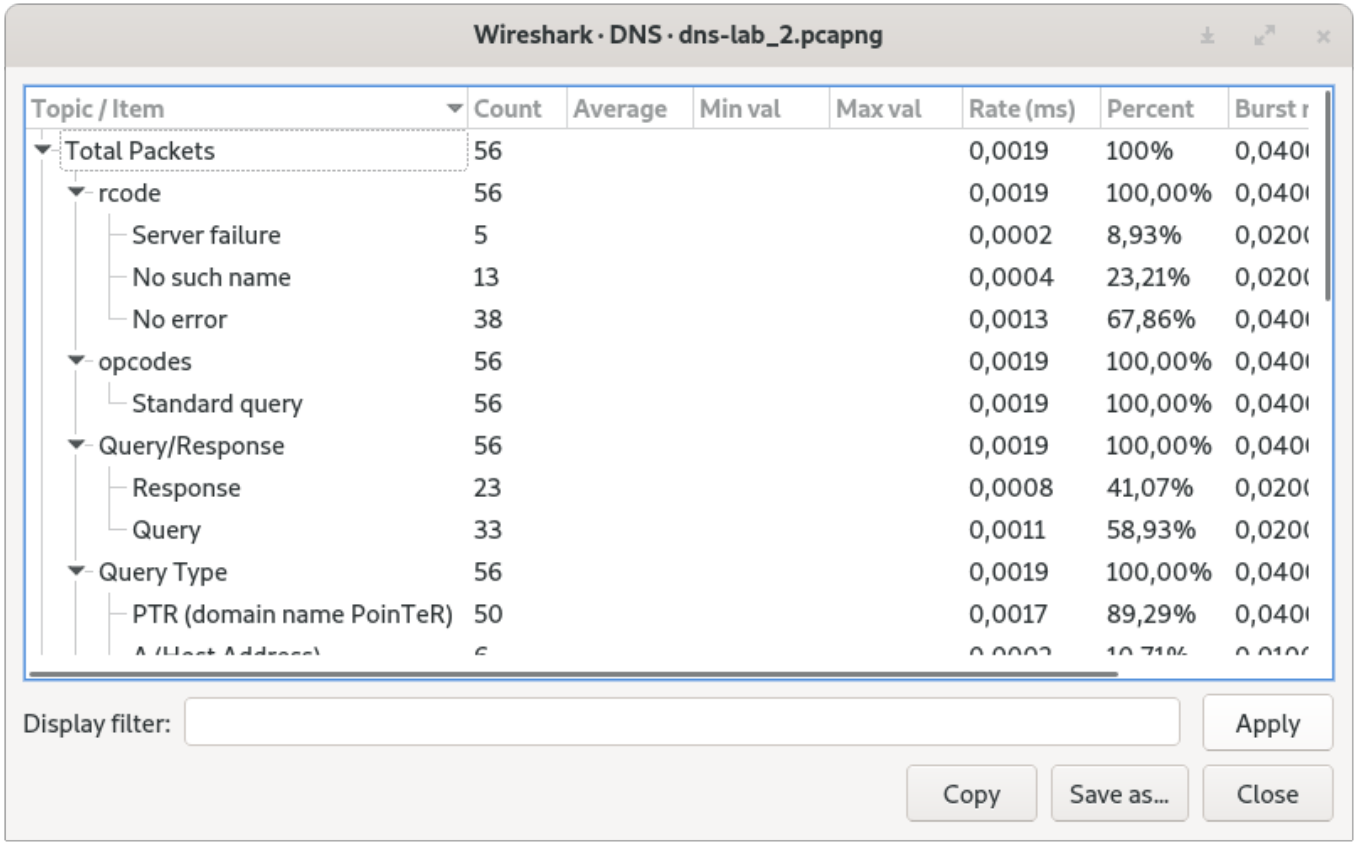
- Formato de los mensajes en wireshark

```
▼ 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

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



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

Capa de aplicación

DNS amenazas y vulnerabilidades

- 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

DNS filtrado

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

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

  - Filtrando por responses

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

  - Si no hay ningun error en la consulta

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

Capa de aplicación

DNS filtrado

- 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.flags.opcode == 1
```

  - 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

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 versiòn

```
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="CAO 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

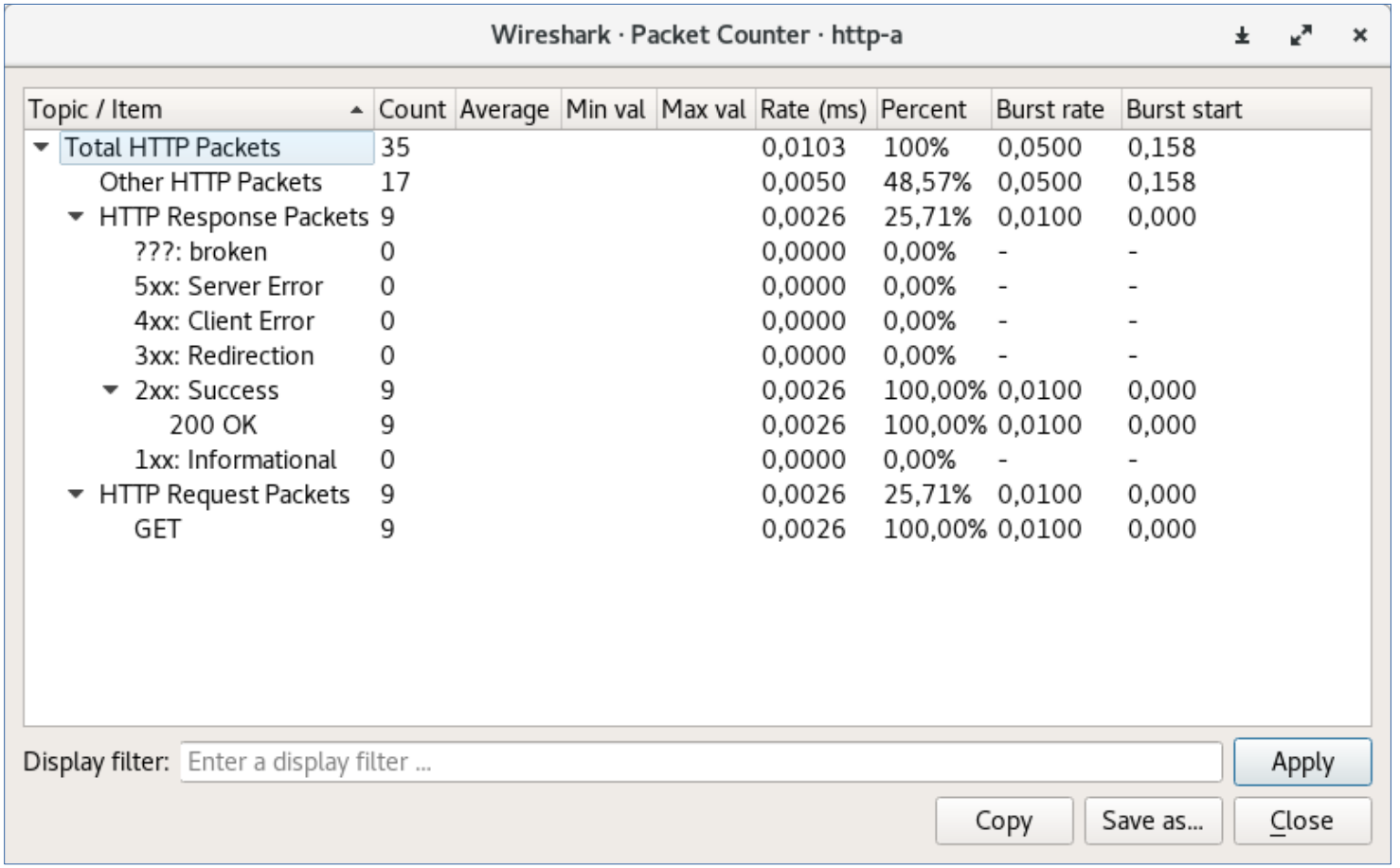
```
http.user_agent matches “[M|m]ozilla”
```

  - Información del host a la que se hace la petición

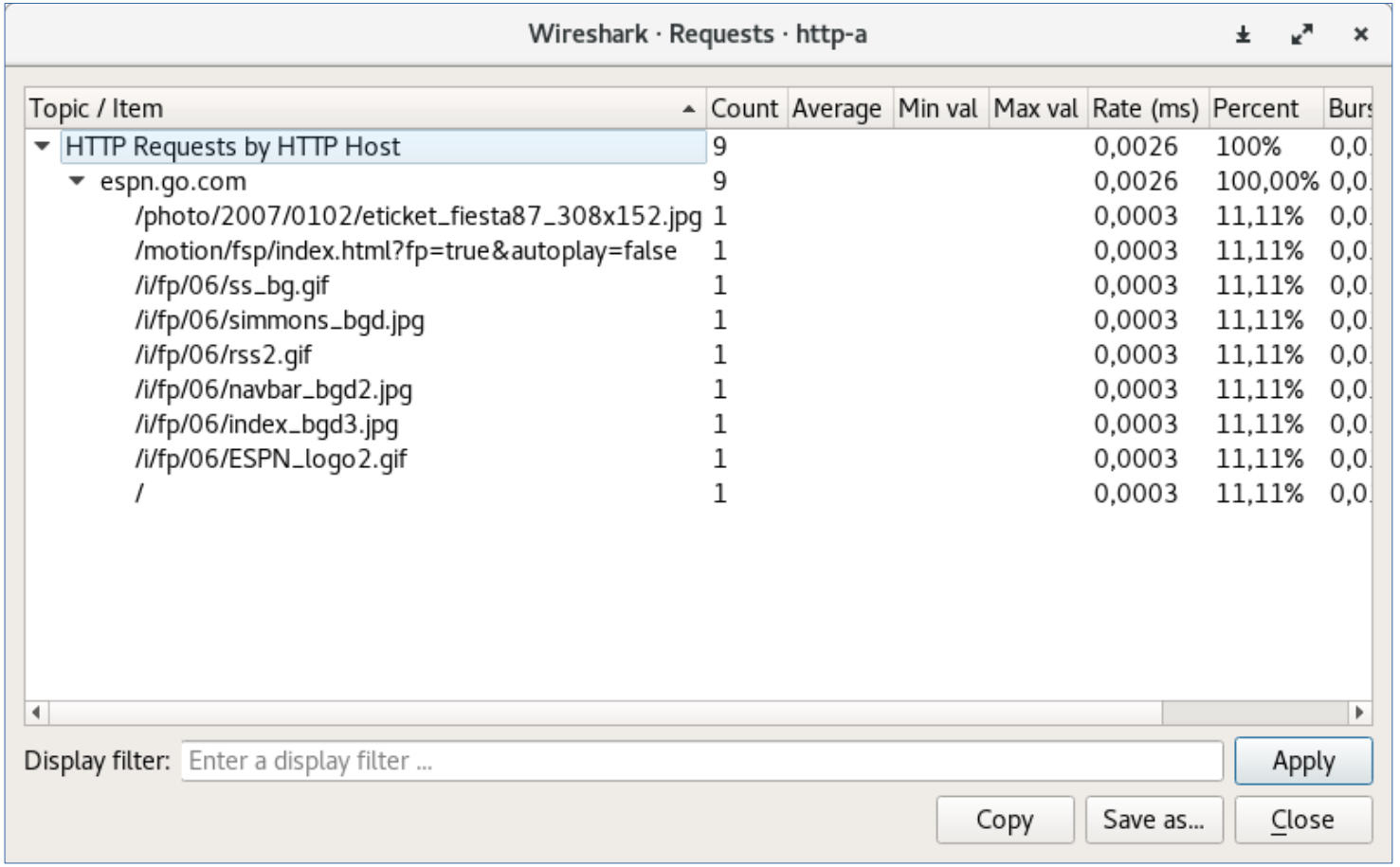
```
http.host contains “hacking”
```

- 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



- Contador de paquetes http
  - Statistics → HTTP → Request



Aplicaciones

Laboratorio 2 - Protocolo HTTP

Archivo: **Fallo HTTP**, Observar el problema de esta traza

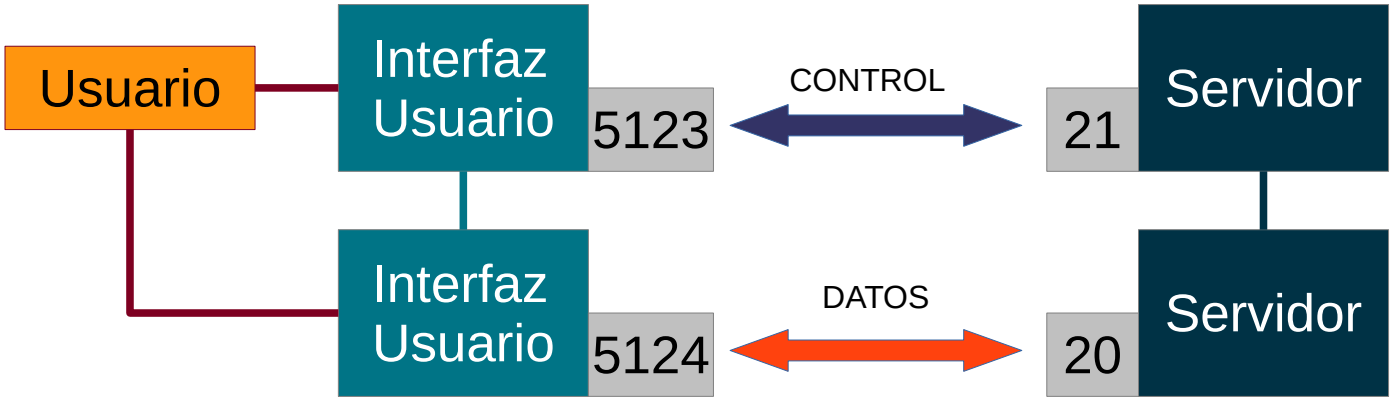
Qué se descarga en el segundo archivo, ¿se puede obtener una copia de la descarga?

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.



- 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
```

- 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