

PRÁCTICA 2

El protocolo HTTP



http://

Índice

Índice	2
EJERCICIO 1: Petición web con cURL	3
EJERCICIO 2: Redirecciones con cURL	5
EJERCICIO 3: Añadiendo encabezados sobre compresión gzip con cURL	9
EJERCICIO 4: Añadiendo encabezados sobre conexiones persistentes con cURL	11
EJERCICIO 5: Averiguar el software que usa un servidor web	13
EJERCICIO 6: Envío de datos en un formulario GET	17
EJERCICIO 7: Envío de datos en un formulario POST	19
EJERCICIO 8: Autenticación básica con cURL	21
EJERCICIO 9: Envío de ficheros con cURL	23
EJERCICIO 10: Búsquedas automatizadas en sitios web	25
EJERCICIO 11: Acceso a servicios web	27
EJERCICIO 12: tcpdump para monitorizar el tráfico de red	29
EJERCICIO 13: ngrep para monitorizar el tráfico de red	30

EJERCICIO 1: Petición web con cURL

Ejecute las acciones indicadas e identifique y describa los mensajes HTTP así como sus diferentes partes (primera línea, encabezados, etc.).

```
adrianacosa@m1-adrianacosa:~$ curl http://void.ugr.es/tweb/hola.html
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>Hola mundo</title>
</head>
<body>
  <h1>¡ Hola, mundo !</h1>
</body>
```

En esta primera ejecución podemos ver que ejecutando el comando cURL sin ninguna opción lo que se realiza es una ejecución de una petición al servidor void.ugr.es de tipo GET para conseguir el archivo que se encuentra en la carpeta tweb/hola.html. En este caso sólo se devuelve por pantalla el resultado de la petición, que en este caso es el documento html ya comentado.

```
adrianacosa@m1-adrianacosa:~$ curl -v http://void.ugr.es/tweb/hola.html
*   Trying 150.214.190.100:80...
* TCP_NODELAY set
* Connected to void.ugr.es (150.214.190.100) port 80 (#0)
> GET /tweb/hola.html HTTP/1.1
> Host: void.ugr.es
> User-Agent: curl/7.68.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Mon, 14 Mar 2022 21:11:27 GMT
< Server: Apache/2.4.46 (Ubuntu)
< Last-Modified: Thu, 03 Jun 2021 16:07:23 GMT
< ETag: "97-5c3dec54ed9f3"
< Accept-Ranges: bytes
< Content-Length: 151
< Vary: Accept-Encoding
< Content-Type: text/html
<
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>Hola mundo</title>
</head>
<body>
  <h1>¡ Hola, mundo !</h1>
</body>
* Connection #0 to host void.ugr.es left intact
```

En esta segunda ejecución, en la que ejecutamos el mismo comando pero con la opción `-v` o *verbose* nos muestra además del resultado de la petición todos los mensajes HTTP que se realizan al momento del establecimiento de la conexión TCP entre el cliente y el servidor. En este caso podemos ver:

- En la primera línea como se intenta la conexión hacia la IP del servidor con destino al puerto 80 (HTTP).
- En la segunda línea se establece el tipo de algoritmo que se usará para realizar la conexión TCP.
- En la tercera la confirmación del establecimiento de la conexión al servidor.
- En la cuarta nos aparece la petición HTTP que hacemos y qué versión del protocolo se usa en dicha petición (en este caso se hace una petición GET y se usa la versión 1.1 de HTTP).
- En la quinta nos muestra el servidor al que se realiza la petición.
- En la sexta aparece el agente de usuario (programa ejecutado en el cliente para realizar la petición) con el que se realiza la petición.
- En la séptima nos sale un mensaje donde se indica la aceptación de la petición.
- En la novena sale el código del éxito en la petición (código 200 OK).
- Y en las siguientes líneas se nos proporciona información relacionada con la petición del tipo de la fecha en la que se realizó, el tipo de servidor al que se realizó y sistema operativo, última vez que se modificó el archivo que se quiere obtener, el ETag que representa al archivo únicamente, la longitud del archivo obtenido y el tipo de contenido que se mostrará.
- Y finalmente se nos muestra el documento HTML que se solicitó por parte del cliente.

EJERCICIO 2: Redirecciones con cURL

Abra un navegador y escriba la URL `http://void.ugr.es` (observe que se ha escrito `http` y no `https`). Podrá ver una página típica por defecto de un servidor Apache y si observa la URL verá que el esquema es `HTTPS` (aunque usted solicitó `HTTP`). Si habilita la herramienta de análisis de tráfico `HTTP` del navegador verá que se han hecho dos peticiones. Esto ocurre porque en `void.ugr.es`, si se recibe una petición con el esquema `HTTP`, es redirigida automáticamente para que use el esquema `HTTPS`. Esto es transparente al usuario del navegador que lo gestiona de forma automática.

En caso de usar `cURL`, podrá comprobar que la redirección no se hace de forma automática.

Ejecute:

```
curl -v http://void.ugr.es
```

Para que `cURL` haga también las redirecciones si fuesen necesarias debe usar el modificador `-L`:

```
curl -vL http://void.ugr.es
```

Ejecute las acciones indicadas con `cURL` y muestre los resultados. A continuación, hágalo también con el navegador e incluya capturas de pantalla de la herramienta de análisis de tráfico de red.

```
adrianacosa@m1-adrianacosa:~$ curl -v http://void.ugr.es
*   Trying 150.214.190.100:80...
* TCP_NODELAY set
* Connected to void.ugr.es (150.214.190.100) port 80 (#0)
> GET / HTTP/1.1
> Host: void.ugr.es
> User-Agent: curl/7.68.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 301 Moved Permanently
< Date: Mon, 14 Mar 2022 21:34:41 GMT
< Server: Apache/2.4.46 (Ubuntu)
< Location: https://void.ugr.es/
< Content-Length: 305
< Content-Type: text/html; charset=iso-8859-1
<
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>301 Moved Permanently</title>
</head><body>
<h1>Moved Permanently</h1>
<p>The document has moved <a href="https://void.ugr.es/">here</a>.</p>
<hr>
<address>Apache/2.4.46 (Ubuntu) Server at void.ugr.es Port 80</address>
</body></html>
* Connection #0 to host void.ugr.es left intact
adrianacosa@m1-adrianacosa:~$
```

En esta ejecución de cURL podemos ver cómo el mensaje HTTP que nos da como respuesta no es 200 como antes, si no que nos devuelve un código 301 Moved Permanently. En este caso se indica que en esa dirección se encontraba anteriormente el archivo que andamos buscando, pero se ha reubicado éste y debemos usar la nueva URL en la que se encuentra dicho archivo, que viene indicada en el apartado *Location*. Esto sucede porque, como dice el enunciado, cURL no sigue las redirecciones de URL a no ser que se le especifique explícitamente. Vamos a ver qué mensajes del protocolo HTTP muestra cURL si le indicamos la opción *-L* para seguir las redirecciones:

```

adrianacosa@1-adrianacosa:~$ curl -vL http://void.ugr.es
*   Trying 150.214.190.100:80...
* TCP_NODELAY set
* Connected to void.ugr.es (150.214.190.100) port 80 (#0)
> GET / HTTP/1.1
> Host: void.ugr.es
> User-Agent: curl/7.68.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 301 Moved Permanently
< Date: Mon, 14 Mar 2022 21:36:04 GMT
< Server: Apache/2.4.46 (Ubuntu)
< Location: https://void.ugr.es/
< Content-Length: 305
< Content-Type: text/html; charset=iso-8859-1
<
* Ignoring the response-body
* Connection #0 to host void.ugr.es left intact
* Issue another request to this URL: 'https://void.ugr.es/'
* Trying 150.214.190.100:443...
* TCP_NODELAY set
* Connected to void.ugr.es (150.214.190.100) port 443 (#1)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
*   CAfile: /etc/ssl/certs/ca-certificates.crt
*   CApath: /etc/ssl/certs
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.3 (IN), TLS handshake, Encrypted Extensions (8):
* TLSv1.3 (IN), TLS handshake, Certificate (11):
* TLSv1.3 (IN), TLS handshake, CERT verify (15):
* TLSv1.3 (IN), TLS handshake, Finished (20):
* TLSv1.3 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.3 (OUT), TLS handshake, Finished (20):
* SSL connection using TLSv1.3 / TLS_AES_256_GCM_SHA384
* ALPN, server accepted to use http/1.1
* Server certificate:
*   subject: CN=void.ugr.es
*   start date: Feb 1 20:53:08 2022 GMT
*   expire date: May 2 20:53:07 2022 GMT
*   subjectAltName: host "void.ugr.es" matched cert's "void.ugr.es"
*   issuer: C=US; O=Let's Encrypt; CN=R3
*   SSL certificate verify ok.
> GET / HTTP/1.1
> GET / HTTP/1.1
> Host: void.ugr.es
> User-Agent: curl/7.68.0
> Accept: */*
>
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* old SSL session ID is stale, removing
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Mon, 14 Mar 2022 21:36:04 GMT
< Server: Apache/2.4.46 (Ubuntu)
< Last-Modified: Wed, 13 Feb 2019 17:37:01 GMT
< ETag: "2aa6-581c9fd4acee3"
< Accept-Ranges: bytes
< Content-Length: 10918
< Vary: Accept-Encoding
< Content-Type: text/html
<
```

Como podemos comprobar, al indicarle que siga las redirecciones, en primer lugar nos muestra el mismo código 301 Moved Permanently. Pero en este caso la cosa no queda ahí. Se realiza una petición GET nueva hacia la URL indicada en el campo *Location* y en este caso al ser una dirección que usa encriptación mediante SSL, se muestran los mensajes de *handshake* donde se negocia la clave de la comunicación entre el cliente y el servidor. Una vez se ha establecido la clave, se procede a obtener la página y a comprobar que somos nosotros mediante el certificado negociado en el paso anterior. Y ahora sí podemos ver cómo el código esta vez es un código 200 OK. Luego la redirección se ha realizado correctamente.

The screenshot shows the Firefox Network Monitor tool. The address bar indicates the URL is https://void.ugr.es. The main pane displays a list of network requests. The first request, a GET for the root URL, has a status of 301 and is highlighted. A detailed view of this 301 response is shown on the right, including the status line (HTTP/1.1 301 Moved Permanently), headers (Date, Server, Location, Content-Length, Keep-Alive, Connection, Content-Type), and request headers (GET / HTTP/1.1, Host, User-Agent, Accept, Accept-Language, Accept-Encoding, Connection, Upgrade-Insecure-Requests). Below the list, a summary bar shows 4 requests, 27.15 KB transferred, and a finish time of 170 ms. The bottom navigation bar includes tabs for Errors, Warnings, Logs, Info, and Debug.

En el caso de usar la herramienta de desarrollador de firefox podemos comprobar cómo se realiza la redirección con dos peticiones HTTP distintas.

- En la primera es en la que nos devuelve el código 301 Moved Permanently y la localización donde realmente se encuentra el fichero de la petición.
- En la segunda es donde se da la confirmación del GET correcto con el código 200 OK indicando que se ha realizado la petición correctamente.

EJERCICIO 3: Añadiendo encabezados sobre compresión gzip con cURL

```
adrianacosa@m1-adrianacosa:~$ curl -v -H "Accept-Encoding: gzip" http://void.ugr.es/tweb/hola.html
*   Trying 150.214.190.100:80...
* TCP_NODELAY set
* Connected to void.ugr.es (150.214.190.100) port 80 (#0)
> GET /tweb/hola.html HTTP/1.1
> Host: void.ugr.es
> User-Agent: curl/7.68.0
> Accept: /*
> Accept-Encoding: gzip
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Mon, 14 Mar 2022 21:56:27 GMT
< Server: Apache/2.4.46 (Ubuntu)
< Last-Modified: Thu, 03 Jun 2021 16:07:23 GMT
< ETag: "97-5c3dec54ed9f3-gzip"
< Accept-Ranges: bytes
< Vary: Accept-Encoding
< Content-Encoding: gzip
< Content-Length: 136
< Content-Type: text/html
<
Warning: Binary output can mess up your terminal. Use "--output -" to tell
Warning: curl to output it to your terminal anyway, or consider "--output
Warning: <FILE>" to save to a file.
* Failed writing body (0 != 136)
* Closing connection 0
```

```
adrianacosa@m1-adrianacosa:~$ curl -v -H "Accept-Encoding: gzip" http://void.ugr.es/tweb/hola.html --output fichero.html.gz
% Total    % Received % Xferd  Average Speed   Time     Time      Time  Current
          Dload  Upload Total Spent   Left Speed
0       0     0      0      0       0      0 ---:---:---:---:---:---:---:--- 0*   Trying 150.214.190.100:80...
* TCP_NODELAY set
* Connected to void.ugr.es (150.214.190.100) port 80 (#0)
> GET /tweb/hola.html HTTP/1.1
> Host: void.ugr.es
> User-Agent: curl/7.68.0
> Accept: /*
> Accept-Encoding: gzip
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Mon, 14 Mar 2022 21:56:55 GMT
< Server: Apache/2.4.46 (Ubuntu)
< Last-Modified: Thu, 03 Jun 2021 16:07:23 GMT
< ETag: "97-5c3dec54ed9f3-gzip"
< Accept-Ranges: bytes
< Vary: Accept-Encoding
< Content-Encoding: gzip
< Content-Length: 136
< Content-Type: text/html
<
{ [136 bytes data]
100 136 100 136 0 0 1307 0 ---:---:---:---:---:---:---:--- 1307
* Connection #0 to host void.ugr.es left intact
adrianacosa@m1-adrianacosa:~$
```

```
adrianacosa@m1-adrianacosa:~$ curl -v --compressed http://void.ugr.es/tweb/hola.html
*   Trying 150.214.190.100:80...
* TCP_NODELAY set
* Connected to void.ugr.es (150.214.190.100) port 80 (#0)
> GET /tweb/hola.html HTTP/1.1
> Host: void.ugr.es
> User-Agent: curl/7.68.0
> Accept: /*
> Accept-Encoding: deflate, gzip, br
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Mon, 14 Mar 2022 21:57:55 GMT
< Server: Apache/2.4.46 (Ubuntu)
< Last-Modified: Thu, 03 Jun 2021 16:07:23 GMT
< ETag: "97-5c3dec54ed9f3-gzip"
< Accept-Ranges: bytes
< Vary: Accept-Encoding
< Content-Encoding: gzip
< Content-Length: 136
< Content-Type: text/html
<
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>Hola mundo</title>
</head>
<body>
  <h1>¡ Hola, mundo !</h1>
</body>
* Connection #0 to host void.ugr.es left intact
```

EJERCICIO 4: Añadiendo encabezados sobre conexiones persistentes con cURL

```
adrianacosa@m1-adrianacosa:~$ curl -v http://void.ugr.es/tweb/hola.html http://void.ugr.es/tweb/hola.html
*   Trying 150.214.190.100:80...
* TCP_NODELAY set
* Connected to void.ugr.es (150.214.190.100) port 80 (#0)
> GET /tweb/hola.html HTTP/1.1
> Host: void.ugr.es
> User-Agent: curl/7.68.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Mon, 14 Mar 2022 22:07:22 GMT
< Server: Apache/2.4.46 (Ubuntu)
< Last-Modified: Thu, 03 Jun 2021 16:07:23 GMT
< ETag: "97-5c3dec54ed9f3"
< Accept-Ranges: bytes
< Content-Length: 151
< Vary: Accept-Encoding
< Content-Type: text/html
<
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>Hola mundo</title>
</head>
<body>
  <h1>¡ Hola, mundo !</h1>
</body>
* Connection #0 to host void.ugr.es left intact
</html>* Found bundle for host void.ugr.es: 0x5614ddd87b70 [seriously]
* Can not multiplex, even if we wanted to!
* Re-using existing connection! (#0) with host void.ugr.es
* Connected to void.ugr.es (150.214.190.100) port 80 (#0)
> GET /tweb/hola.html HTTP/1.1
> Host: void.ugr.es
> User-Agent: curl/7.68.0
> Accept: */*
>
```

El resultado de la ejecución de este comando donde realizamos la misma petición sobre el mismo servidor nos da como resultado un mensaje en medio que dice lo siguiente: *"Re-using existing connection! (#0) with host void.ugr.es"*. Esto nos indica que se reutiliza la conexión TCP/IP anterior para realizar otra petición al mismo servidor. Ahora vamos a añadir la cabecera *Connection: Close* para comprobar si cierra la conexión y vuelve a abrir otra conexión nueva TCP para volver a conectarse al mismo servidor:

```
adrianacosa@m1-adrianacosa:~$ curl -v -H "Connection: Close" http://void.ugr.es/tweb/hola.html http://void.ugr.es/tweb/hola.html
*   Trying 150.214.190.100:80...
* TCP_NODELAY set
* Connected to void.ugr.es (150.214.190.100) port 80 (#0)
> GET /tweb/hola.html HTTP/1.1
> Host: void.ugr.es
> User-Agent: curl/7.68.0
> Accept: */*
> Connection: Close
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Mon, 14 Mar 2022 22:13:01 GMT
< Server: Apache/2.4.46 (Ubuntu)
< Last-Modified: Thu, 03 Jun 2021 16:07:23 GMT
< ETag: "97-5c3dec54ed9f3"
< Accept-Ranges: bytes
< Content-Length: 151
< Vary: Accept-Encoding
< Connection: close
< Content-Type: text/html
<
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>Hola mundo</title>
</head>
<body>
  <h1>¡Hola, mundo !</h1>
</body>
* Closing connection 0
</html>* Hostname void.ugr.es was found in DNS cache
* Trying 150.214.190.100:80...
* TCP_NODELAY set
* Connected to void.ugr.es (150.214.190.100) port 80 (#1)
> GET /tweb/hola.html HTTP/1.1
> Host: void.ugr.es
> User-Agent: curl/7.68.0
> Accept: */*
> Connection: Close
>
```

En esta nueva petición indicando que cierre la conexión tras terminar la respuesta a la petición podemos comprobar que cuando el servidor responde con el archivo pedido por el cliente la conexión se cierra. En el momento en el que volvemos a realizar otra petición al mismo servidor, en este caso no se reutiliza la conexión anterior, si no que se genera una nueva. Nos damos cuenta en que el número de la conexión pasa de #0 a #1, indicando claramente que son dos conexiones distintas hacia el mismo servidor. Si usamos la cabecera “*Connection: Keep-Alive*” obtenemos el comportamiento por defecto de cURL si no le indicamos la cabecera.

EJERCICIO 5: Averiguar el software que usa un servidor web

Averigüe qué servidor web utiliza cada uno de los sitios del siguiente listado. Tenga en cuenta que es posible usar esquemas HTTP o HTTPS y que algunos hacen redirectiones:

- **void.ugr.es:** Apache Web Server

```
* Trying 150.214.190.100:80...
* TCP_NODELAY set
* Connected to void.ugr.es (150.214.190.100) port 80 (#0)
> GET / HTTP/1.1
> Host: void.ugr.es
> User-Agent: curl/7.68.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 301 Moved Permanently
< Date: Mon, 14 Mar 2022 22:24:19 GMT
< Server: Apache/2.4.46 (Ubuntu)
< Location: https://void.ugr.es/
< Content-Length: 305
< Content-Type: text/html; charset=iso-8859-1
```

-
- www.ugr.es: Nginx Web Server

```
* Trying 150.214.27.71:80...
* TCP_NODELAY set
* Connected to www.ugr.es (150.214.27.71) port 80 (#0)
> GET / HTTP/1.1
> Host: www.ugr.es
> User-Agent: curl/7.68.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 301 Moved Permanently
< Server: nginx
< Date: Mon, 14 Mar 2022 22:25:45 GMT
< Content-Type: text/html
< Content-Length: 162
< Connection: keep-alive
< Location: https://www.ugr.es/
```

- www.lamoncloa.gob.es: Microsoft IIS Web Server

```
* Mark bundle as not supporting multiuse
< HTTP/1.1 302 Redirect
< Content-Type: text/html; charset=UTF-8
< Location: https://www.lamoncloa.gob.es/Paginas/index.aspx
< Server: Microsoft-IIS/8.0
< X-SharePointHealthScore: 0
< SPRequestGuid: f5cb29a0-92b8-b025-e01d-c502374ea772
< request-id: f5cb29a0-92b8-b025-e01d-c502374ea772
< X-FRAME-OPTIONS: SAMEORIGIN
< SPRequestDuration: 13
< SPIisLatency: 0
< X-Powered-By: ASP.NET
< X-Content-Type-Options: nosniff
< X-MS-InvokeApp: 1; RequireReadOnly
< Date: Mon, 14 Mar 2022 22:35:31 GMT
< Content-Length: 169
< Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
<
<head><title>Document Moved</title></head>
* Connection #0 to host www.lamoncloa.gob.es left intact
```

-
- **es.wikipedia.org:** Varnish HTTP Cache

```
* Trying 91.198.174.192:80...
* TCP_NODELAY set
* Connected to es.wikipedia.org (91.198.174.192) port 80 (#0)
> GET / HTTP/1.1
> Host: es.wikipedia.org
> User-Agent: curl/7.68.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 301 TLS Redirect
< Date: Mon, 14 Mar 2022 22:36:46 GMT
< Server: Varnish
```

- **www.elmundo.es:** Varnish HTTP Cache

```
aulas@lubuntu:~$ curl -v www.elmundo.es | head -n10
% Total    % Received % Xferd  Average Speed   Time     Time      Time  Current
               Dload  Upload   Total   Spent    Left  Speed
0       0     0       0     0       0       0       0 --::-- --::-- --::-- 0*
* TCP_NODELAY set
* Connected to www.elmundo.es (199.232.193.50) port 80 (#0)
> GET / HTTP/1.1
> Host: www.elmundo.es
> User-Agent: curl/7.68.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 301 Moved Permanently
< Retry-After: 0
< Location: https://www.elmundo.es/
< Content-Length: 0
< Accept-Ranges: bytes
< Date: Fri, 18 Mar 2022 17:40:42 GMT
< Connection: close
< X-Served-By: cache-lcy19244-LCY
< X-Cache: HIT
< X-Cache-Hits: 0
<
0       0     0       0     0       0       0       0 --::-- --::-- --::-- 0
* Closing connection 0
aulas@lubuntu:~$
```

Lo he averiguado buscando el servidor caché que devuelve en el campo x-served-by.

Si buscamos en google encontramos:

-
- www.medium.com: Cloudflare Web Server

```
* Trying 162.159.152.4:80...
* TCP_NODELAY set
* Connected to www.medium.com (162.159.152.4) port 80 (#0)
> GET / HTTP/1.1
> Host: www.medium.com
> User-Agent: curl/7.68.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 301 Moved Permanently
< Date: Mon, 14 Mar 2022 22:49:05 GMT
< Transfer-Encoding: chunked
< Connection: keep-alive
< Cache-Control: max-age=3600
< Expires: Mon, 14 Mar 2022 23:49:05 GMT
< Location: https://www.medium.com/
< X-Content-Type-Options: nosniff
< Set-Cookie: __cftruid=df0b7a0ef0653ef3b89a1d00a2913a7d4b2652ce-1647298145; path=/; domain=.medium.com; HttpOnly
< Server: cloudflare
< CF-RAY: 6ec08f80d8778678-MAD
< alt-svc: h3=":443"; ma=86400, h3-29=:443; ma=86400
<
* Connection #0 to host www.medium.com left intact
```

- www.slack.com: Envoy Proxy

```
* Trying 3.68.170.153:80...
* TCP_NODELAY set
* Connected to www.slack.com (3.68.170.153) port 80 (#0)
> GET / HTTP/1.1
> Host: www.slack.com
> User-Agent: curl/7.68.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 301 Moved Permanently
< location: https://www.slack.com/
< date: Mon, 14 Mar 2022 22:50:03 GMT
< server: envoy
< content-length: 0
<
* Connection #0 to host www.slack.com left intact
```

-
- www.whitehouse.gov: Nginx Web Server

```
[ [5 bytes data]
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
[ [265 bytes data]
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
[ [265 bytes data]
* old SSL session ID is stale, removing
[ [5 bytes data]
* Connection state changed (MAX_CONCURRENT_STREAMS == 128)!
} [5 bytes data]
< HTTP/2 200
< server: nginx
< date: Mon, 14 Mar 2022 22:51:35 GMT
< content-type: text/html; charset=UTF-8
< content-length: 135308
< x-build-back-better: https://usds.gov/
< x-frame-options: DENY
< referrer-policy: strict-origin-when-cross-origin
< x-xss-protection: 1; mode=block
< x-content-type-options: nosniff
< x-rq: mad2 0 4 9980
< cache-control: max-age=300, must-revalidate
< age: 1700
< x-cache: hit
< vary: Accept-Encoding
< accept-ranges: bytes
< strict-transport-security: max-age=31536000;includeSubdomains;preload
<
[ [1054 bytes data]
100 132k 100 132k 0 0 825k 0 ---:--- ---:--- ---:--- 825k
* Connection #0 to host www.whitehouse.gov left intact
```

EJERCICIO 6: Envío de datos en un formulario GET

Para poder entrar en nuestra carpeta asignada en el servidor void.ugr.es para montarla en una carpeta en nuestro sistema operativo lo hacemos mediante la orden sshfs de la siguiente manera:

```
adri@hacker ~/Documents> sudo sshfs -o allow_other adrianacosa2122@void.
ugr.es:/~/Documents/void_ugr_es/
The authenticity of host 'void.ugr.es (150.214.190.100)' can't be established.
ED25519 key fingerprint is SHA256:ZdkKfb0vN52bSdsHj11IRHMXd005phgTzMo0J14IfU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
adrianacosa2122@void.ugr.es's password:
```

Una vez iniciamos sesión en el servidor, tendremos montada la carpeta home de nuestro usuario en la carpeta especificada como parámetro. En este caso yo le he dado la ruta de mi carpeta Documents y dentro de ésta otra carpeta más llamada void_ugr_es. Una vez montada la carpeta, creamos el fichero especificado en la carpeta public_html.

Ahora procedemos a acceder desde el navegador a dicho archivo y a ver mediante las herramientas de desarrollo lo que pasa cuando rellenamos el formulario.

Formulario

Nombre: adrianacosa2122 Clave: Enviar

https://void.ugr.es/~adrianacosa2122/formulario.html

Una vez llenado el formulario, cuando le damos a enviar nos manda a otra página llamada *procesar.php* donde nos da la información que le hemos proporcionado procesada de la petición GET.

Variables recibidas

Desde \$_GET

- nombre = adrianacosa2122
- password = mKpaZHTU
- enviar = Enviar

Desde \$_POST

Status	Method	Domain	File	Initiator	Type	Transferred	Size
200	GET	void.ugr.es	procesar.php?nombre=adrianacosa2122&password=mKpaZHTU&enviar=Enviar	document	html	526 B	456 B
200	GET	void.ugr.es	favicon.ico	FaviconLoader.jsm:191 (img)	image	5.56 KB	5.56 KB

Headers

GET

Scheme: https
Host: void.ugr.es
Filename: ./procesar.php

nombre: adrianacosa2122
password: mKpaZHTU
enviar: Enviar

Address: 150.214.190.100:443

200 OK

HTTP/1.1

526 B (456 B size)

strict-origin-when-cross-origin

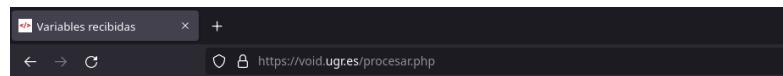
EJERCICIO 7: Envío de datos en un formulario POST

Modifique el formulario y cambie el método de envío a “POST”. A continuación repita el procedimiento del ejercicio anterior desde el navegador. Podrá comprobar que se ha modificado la forma en la que se envían los datos al servidor.

El archivo modificado sería el siguiente:

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Formulario</title>
    </head>
    <body>
        <form action="https://void.ugr.es/procesar.php" method="POST">
            <label>Nombre: <input type="text" name="nombre"></label>
            <label>Clave: <input type="password" name="password"></label>
            <input type="submit" name="enviar" value="Enviar">
        </form>
    </body>
</html>
```

Vamos a ver ahora qué mensajes nos devuelve el formulario y cómo se envía la información con la herramienta de desarrollo del navegador:

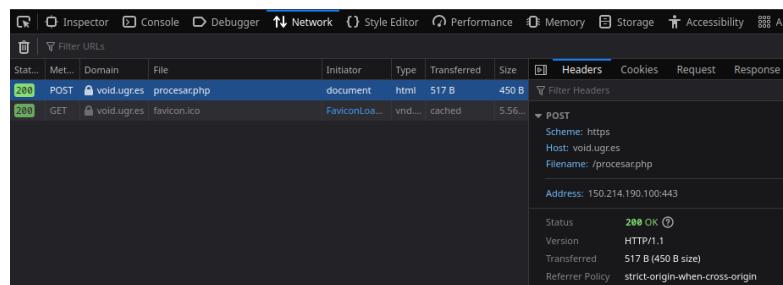


Variables recibidas

Desde \$_GET

Desde \$_POST

- nombre = **pepito**
- password = **contrasenia**
- enviar = **Enviar**



Ahora la información se muestra como parte de las peticiones POST en vez de en las peticiones GET.

También podemos ver que desde cURL no se muestra el cuerpo del mensaje enviado. Como se puede apreciar en la siguiente imagen, sólo aparecen las cabeceras seguidas de una línea en blanco:

```
adri@hacker ~> curl -d "nombre=pepito&password=laClave&enviar=Enviar" -v https://void.ugr.es/procesar.php
* Trying 150.214.190.100:443...
* Connected to void.ugr.es (150.214.190.100) port 443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* CAfile: /etc/ssl/certs/ca-certificates.crt
* Capath: none
TLSV1.3 (OUT), TLS handshake, Client hello (1):
TLSV1.3 (IN), TLS handshake, Server hello (2):
TLSV1.3 (IN), TLS handshake, Encrypted Extensions (8):
TLSV1.3 (IN), TLS handshake, Certificate (11):
TLSV1.3 (IN), TLS handshake, CERT verify (15):
TLSV1.3 (IN), TLS handshake, Finished (20):
TLSV1.3 (OUT), TLS change cipher, Change cipher spec (1):
TLSV1.3 (OUT), TLS handshake, Finished (20):
SSL connection using TLSV1.3 / TLS_AES_256_GCM_SHA384
ALPN, server accepted to use http/1.1
Server certificate:
* subject: CN=void.ugr.es
* start date: Feb 1 20:53:08 2022 GMT
* expire date: May 2 20:53:07 2022 GMT
* subjectAltName: host "void.ugr.es" matched cert's "void.ugr.es"
* issuer: C=US; O=Let's Encrypt; CN=R3
* SSL certificate verify ok.
> POST /procesar.php HTTP/1.1
> Host: void.ugr.es
> User-Agent: curl/7.82.0
> Accept: */
> Content-Length: 44
> Content-Type: application/x-www-form-urlencoded
>
* TLSV1.3 (IN), TLS handshake, Newsession Ticket (4):
* TLSV1.3 (IN), TLS handshake, Newsession Ticket (4):
* old SSL session ID is stale, removing
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Wed, 16 Mar 2022 22:05:11 GMT
< Server: Apache/2.4.46 (Ubuntu)
< Vary: Accept-Encoding
< Content-Length: 446
< Content-Type: text/html; charset=UTF-8
<
<!DOCTYPE html>
<html>
```



```
adri@hacker ~> curl -d "nombre=pepito&password=laClave&enviar=Enviar" --trace-ascii -H "Content-Type: application/x-www-form-urlencoded" https://void.ugr.es/procesar.php
== Info: Trying 150.214.190.100:443...
== Info: Connected to void.ugr.es (150.214.190.100) port 443 (#0)
== Info: ALPN, offering h2
== Info: ALPN, offering http/1.1
== Info: CAfile: /etc/ssl/certs/ca-certificates.crt
== Info: Capath: none
=> Send SSL data, 5 bytes (0x5)
0000: .....
== Info: TLSV1.3 (OUT), TLS handshake, Client hello (1):
=> Send SSL data, 512 bytes (0x200)
0000: ....._c.....%Yr.....%....'$..7h..B.G.8.%vt..%M.\(08..nB|
0040: ..u..I[...>.....,0.....+./..$.(..k.#)'g.....9.....3t.....
0080: <.5./.....u.....void.ugr.es.....3t.....
*00c0: ..h2.http/1.1.....1.....0
0100: .....+.....3.&.$.....z.*v.....)G&..U....
0140: #.Le..D
0180: .....+
01c0: .....+
=< Recv SSL data, 5 bytes (0x5)
0000: .....
== Info: TLSV1.3 (IN), TLS handshake, Server hello (2):
=< Recv SSL data, 122 bytes (0x7a)
0000: .....v..U."05.....z=:.m..y.....[!.% B.G.8.%vt..%M.\(08..nB|
0040: ..u..I[...+.+.....3.S.....).l._m.....m.....x.X.-.4,
=< Recv SSL data, 5 bytes (0x5)
0000: .....
=< Recv SSL data, 5 bytes (0x5)
0000: .....
=< Recv SSL data, 1 bytes (0x1)
0000: .
== Info: TLSV1.3 (IN), TLS handshake, Encrypted Extensions (8):
=< Recv SSL data, 25 bytes (0x19)
0000: .....http/1.1
=< Recv SSL data, 5 bytes (0x5)
0000: .....
=< Recv SSL data, 1 bytes (0x1)
0000: .
== Info: TLSV1.3 (IN), TLS handshake, Certificate (11):
=< Recv SSL data, 4021 bytes (0xfb5)
0000: .....0.....W.o.g.....A.....0...*H.....021.0
0040: ..U.....U$1.0.....U.....Let's Encrypt1.0.....U.....R30.....220201205308Z
0080: ..220502205307Z0.1.0.....U.....void.ugr.es0.."0...*H.....0
00c0: .....g.....o{k.....B...,5..YqL.....*.....#$.k'....?
```

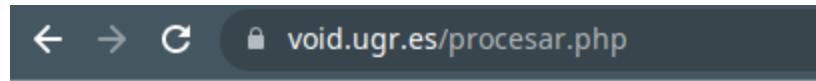
Para poder mostrar esa información hacemos uso de la opción `--trace-ascii` que proporciona más detalle de la información intercambiada.

EJERCICIO 8: Autenticación básica con cURL

Se pide que introduzcamos en nuestro `/home` el archivo `procesar.php` y que en la cláusula `action` de nuestro fichero `formulario.html` le indiquemos éste:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Formulario</title>
  </head>
  <body>
    <form action="https://void.ugr.es/~adrianacosa212/public_html/procesar.php" method="POST">
      <label>Nombre: <input type="text" name="nombre"></label>
      <label>Clave: <input type="password" name="password"></label>
      <input type="submit" name="enviar" value="Enviar">
    </form>
  </body>
</html>
```

Una vez tenemos todo lo necesario, si nos autenticamos en el navegador todo funciona como debería:



Variables recibidas

Desde \$_GET

Desde \$_POST

- nombre = pepito
- password = contrasenia
- enviar = Enviar

Sin embargo, desde `cURL` debemos usar una opción especial que es `--user` de la siguiente manera:

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>401 Unauthorized</title>
</head><body>
<h1>Unauthorized</h1>
<p>This server could not verify that you
are authorized to access the document
requested. Either you supplied the wrong
credentials (e.g., bad password), or your
browser doesn't understand how to supply
the credentials required.</p>
<hr>
<address>Apache/2.4.46 (Ubuntu) Server at void.ugr.es Port 443</address>
</body></html>
```

Mensaje de error que sale cuando no nos autenticamos

```
adri@hacker ~/D/v/public_html> curl --user adrianacosa2122:mKpaZHTU -d "nombre=pepito&password=laclave&enviar=Enviar" -
-trace-ascii - https://void.ugr.es/~adrianacosa2122/procesar.php
```

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Variables recibidas</title>
    <style type="text/css">
      .valor { color: red; }
    </style>
  </head>
  <body>
    <h1>Variables recibidas</h1><h2>Desde $_GET</h2><ul></ul><h2>Desde $_POST</h2><ul><li>nombre = <span class='valor'>pe
pito</span></li><li>password = <span class='valor'>laclave</span></li><li>enviar = <span class='valor'>Enviar</span></l
i></ul></body>
</html>
```

Cuando usamos `--user` para identificarnos.

EJERCICIO 9: Envío de ficheros con cURL

Creamos el fichero *formulario2.html* con el contenido indicado en el guión y lo añadimos a nuestro *public_html*. Para comprobar que funciona, entramos desde el navegador y adjuntamos una imagen:

The screenshot shows a browser window with two tabs. The top tab is titled "Formulario" and displays a form with fields for Nombre (pepito), Clave (*****), and Imagen (Choose File perro.jpg). The bottom tab shows the result of the submission with the title "Desde \$_GET" and "Desde \$_POST". The \$_POST section lists: nombre = pepito, password = grillo, enviar = Enviar. The \$_FILES section shows an array for the uploaded file: Array ([name] => perro.jpg [type] => image/jpeg [tmp_name] => /tmp/phptczx3f [error] => 0 [size] => 170804). Below this, a thumbnail image of a golden retriever dog is displayed.

Desde \$_GET

Desde \$_POST

- nombre = pepito
- password = grillo
- enviar = Enviar

Ficheros recibidos:

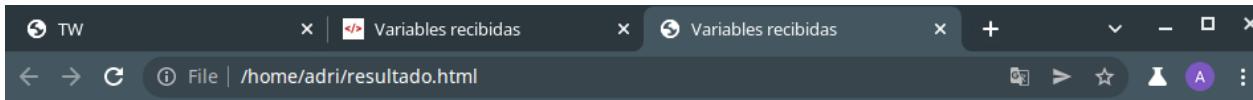
- imagen =

```
Array
(
    [name] => perro.jpg
    [type] => image/jpeg
    [tmp_name] => /tmp/phptczx3f
    [error] => 0
    [size] => 170804
)
```



Y como vemos funciona correctamente. Para enviar ficheros con cURL lo hacemos con la opción `-F` de la siguiente manera:

```
adri@hacker ~ [26]> curl -F "nombre=pepito&password=laclave&enviar=Enviar" -F "imagen=@perro.jpg" https://void.ugr.es/procesar2.php > resultado.html
% Total    % Received % Xferd  Average Speed   Time      Time     Current
          Dload  Upload   Total   Spent   Left  Speed
100  390k    0  223k  100  167k  529k  396k --:--:-- --:--:-- --:--:--  926k
```



Variables recibidas

Desde \$_GET

Desde \$_POST

- nombre = pepito&password=laclave&enviar=Enviar

Ficheros recibidos:

- imagen =

```
Array
(
    [name] => perro.jpg
    [type] => image/jpeg
    [tmp_name] => /tmp/phpWNn6ET
    [error] => 0
    [size] => 170804
)
```



EJERCICIO 10: Búsquedas automatizadas en sitios web

Si hacemos la búsqueda que nos indica el enunciado del ejercicio, google nos devuelve el siguiente mensaje:

403. That's an error.

Your client does not have permission to get URL /search?
q=gatitos from this server. (Client IP address:
85.136.41.180)

Please see Google's Terms of Service posted at
<https://policies.google.com/terms>

If you believe that you have received this response in error, please [report](#) your problem. However, please make sure to take a look at our Terms of Service (http://www.google.com/terms_of_service.html). In your email, please send us the **entire** code displayed below. Please also send us any information you may know about how you are performing your Google searches-- for example, "I'm using the Opera browser on Linux to do searches from home. My Internet access is through a dial-up account I have with the FooCorp ISP." or "I'm using the Konqueror browser on Linux to search from my job at myFoo.com. My machine's IP address is 10.20.30.40, but all of myFoo's web traffic goes through some kind of proxy server whose IP address is 10.11.12.13." (If you don't know any information like this, that's OK. But this kind of information can help us track down problems, so please tell us what you can.)

We will use all this information to diagnose the problem, and we'll hopefully have you back up and searching with Google again quickly!

Please note that although we read all the email we receive, we are not always able to send a personal response to each and every email. So don't despair if you don't hear back from us!

Also note that if you do not send us the **entire** code below, *we will not be able to help you*.

Best wishes,
The Google Team

Esto sucede debido a que google bloquea las búsquedas automatizadas y detecta la búsqueda con *cURL* como tal. Existen herramientas hechas por la comunidad de Linux que permiten hacer este tipo de búsquedas sin que google pueda identificar nuestra búsqueda con *cURL* como una búsqueda automatizada.

Podemos comprobar fácilmente que podemos hacer ésta misma búsqueda sobre la página de la Biblioteca del Congreso de los Estados Unidos, ya que ésta no bloquea este tipo de búsquedas:

The screenshot shows a web browser window with the title bar "Search results for Gatitos, Av". The address bar shows the URL "/home/adri/congreso_usa.html". The page content includes a "Top of page" link, a "Skip to main content" link, and a search bar with the text "gatitos". Below the search bar, there's a "Library of Congress » Search" link and a "Share" button. The search results are titled "Results: 1-25 of 29 | Refined by: Available Online [Remove](#)". There are two main sections of results:

- Audio Recording**
Zape, gatito
 - Contributor: Lomax, Ruby T. (Ruby Terrill) - Lomax, John A. (John Avery) - Longoria, Manuela
 - Date: 1939-04-25
- Audio Recording**
Mi gatito
 - Contributor: Lomax, Ruby T. (Ruby Terrill) - Lomax, John A. (John Avery) - Salazar, Isabella
 - Date: 1939-05-02
- Manuscript/Mixed Material**
Disc sleeve for AFS Disc #2608
 - Contributor: Lomax, Ruby T. (Ruby Terrill)
 - Date: 1939-04-25

A "Back to top" button is located at the bottom right of the page.

EJERCICIO 11: Acceso a servicios web

api_key *

(string) *
5b3ce3597851110001cf62488305ac28ad91466881890. [X](#) [?](#)

start *

(string) eg.: 8.681495,49.41461*
-3.6130750179290776, 37.186441837394554 [?](#)

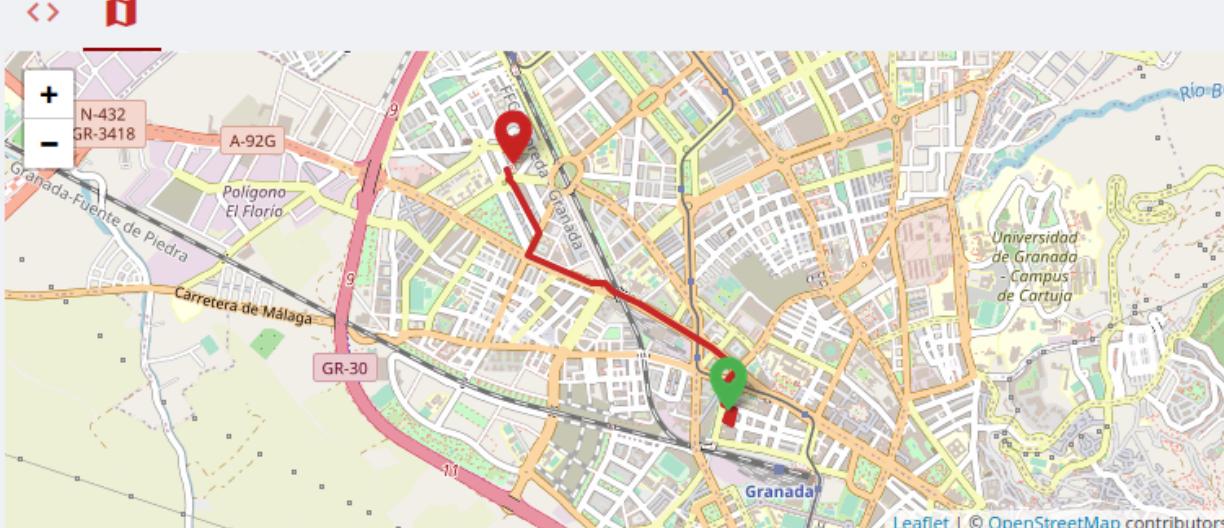
end *

(string) eg.: 8.687872,49.420318*
-3.624405, 37.196839 [?](#)



[SHOW EXAMPLE CODE](#) [CALL ACTION](#)

API response:


Leaflet | © OpenStreetMap contributors

[DOWNLOAD](#)

```
adr1@hacker ~> curl --insecure \
    --header "Content-Type: application/json; charset=utf-8" \
    --header "Accept: application/json, application/geo+json, application/gpx+xml, img/png; charset=utf-8" \
    'https://api.openrouteservice.org/v2/directions/driving-car?api_key=5b3ce3597851110001cf62488305ac28ad914668818903e4900b17de&start=-3.6130750179290776,%2037.186441837394554&end=-3.624405,%2037.196839%20'
HTTP/2 200
server: nginx/1.21.4
date: Fri, 18 Mar 2022 11:22:33 GMT
content-type: application/geojson; charset=UTF-8
access-control-allow-origin: *
access-control-expose-headers: Access-Control-Allow-Origin,Access-Control-Allow-Credentials
x-ratelimit-limit: 2000
x-ratelimit-remaining: 1977
x-ratelimit-reset: 1647687782
access-control-allow-methods: OPTIONS,GET,POST
access-control-allow-headers: Access-Control-Allow-Headers, Origin,Accept, X-Requested-With, Authorization, Content-Type, Access-Control-Request-Method, Access-Control-Request-Headers
access-control-expose-headers: x-ratelimit-remaining, x-ratelimit-reset, x-ratelimit-limit

{"type": "FeatureCollection", "features": [{"bbox": [-3.624742, 37.18598, -3.612566, 37.196715], "type": "Feature", "properties": {"segments": [{"distance": 1984.6, "duration": 281.5, "steps": [{"distance": 45.8, "duration": 11.0, "type": 11, "instruction": "Head south on Calle Azor", "name": "Calle Azor", "way_points": [0, 2]}, {"distance": 28.5, "duration": 4.1, "type": 0, "instruction": "Turn left onto Calle T\u00f3rtola", "name": "Calle T\u00f3rtola", "way_points": [2, 3]}, {"distance": 73.3, "duration": 17.6, "type": 0, "instruction": "Turn left onto Calle Alondra", "name": "Calle Alondra", "way_points": [10, 19]}, {"distance": 233.7, "duration": 32.0, "type": 7, "instruction": "Turn right onto Calle Mirlo", "name": "Calle Mirlo", "way_points": [19, 37]}, {"distance": 233.9, "duration": 32.0, "type": 7, "instruction": "Turn left onto Avenida de Andalucia", "name": "Avenida de Andalucia", "way_points": [19, 37]}, {"distance": 48.9, "duration": 8.8, "type": 6, "instruction": "Enter the roundabout and take the 2nd exit onto Avenida de Andalucia", "name": "Avenida de Andalucia", "exit_number": 2, "way_points": [37, 49]}, {"distance": 124.4, "duration": 17.8, "type": 6, "instruction": "Continue straight onto Avenida de Andalucia", "name": "Avenida de Andalucia", "way_points": [51, 55]}, {"distance": 19.0, "duration": 19.2, "type": 1, "instruction": "Turn right onto Calle Periodista Jos\u00e9 Mar\u00eda Carulla", "name": "Calle Periodista Jos\u00e9 Mar\u00eda Carulla", "way_points": [55, 63]}, {"distance": 276.1, "duration": 39.8, "type": 0, "instruction": "Turn left onto Calle Periodista Rafael Gago Palomo", "name": "Calle Periodista Rafael Gago Palomo", "way_points": [63, 68]}, {"distance": 66.6, "duration": 12.7, "type": 7, "instruction": "Enter the roundabout and take the 2nd exit onto Calle Periodista Daniel Saucedo Aranda", "name": "Calle Periodista Daniel Saucedo Aranda", "exit_number": 2, "way_points": [68, 75]}, {"distance": 0.0, "duration": 10, "type": 10, "instruction": "Arrive at Calle Periodista Daniel Saucedo Aranda, on the right", "name": "-", "way_points": [75, 75]}, {"summary": {"distance": 1984.6, "duration": 281.5}, "geometry": {"coordinates": [[-3.613073, 37.186441], [-3.613192, 37.186082], [-3.613205, 37.186043], [-3.612894, 37.18598], [-3.612881, 37.186017], [-3.612698, 37.186573], [-3.612683, 37.186618], [-3.612985, 37.186705], [-3.613282, 37.186791], [-3.613306, 37.186798], [-3.613327, 37.186804], [-3.613091, 37.187636], [-3.613079, 37.187677], [-3.613072, 37.187701], [-3.613059, 37.187754], [-3.613044, 37.187844], [-3.613028, 37.187889], [-3.613012, 37.187904], [-3.612707, 37.188177], [-3.612566, 37.188285], [-3.612698, 37.188374], [-3.612729, 37.188395], [-3.613864, 37.18916], [-3.614394, 37.189475], [-3.615006, 37.189759], [-3.6155, 37.18989], [-3.615986, 37.190209], [-3.616577, 37.190509], [-3.61678, 37.190628], [-3.617664, 37.191029], [-3.617843, 37.191111], [-3.617989, 37.191177], [-3.618258, 37.19129], [-3.6187, 37.191452], [-3.618891, 37.191522], [-3.619101, 37.191599], [-3.619373, 37.191713], [-3.619442, 37.191752], [-3.619461, 37.191801], [-3.619518, 37.191861], [-3.619575, 37.191892], [-3.619699, 37.191913], [-3.619774, 37.191983], [-3.61984, 37.191875], [-3.620015, 37.191893], [-3.620278, 37.191946], [-3.620948, 37.192152], [-3.621625, 37.192371], [-3.621803, 37.192429], [-3.621845, 37.192441], [-3.621959, 37.19247], [-3.622359, 37.192603], [-3.622607, 37.192687], [-3.622773, 37.192764], [-3.623412, 37.193009], [-3.62362, 37.193093], [-3.623602, 37.193126], [-3.623551, 37.193226], [-3.623502, 37.193257], [-3.623447, 37.193349], [-3.623311, 37.193577], [-3.623252, 37.193675], [-3.623115, 37.193905], [-3.623027, 37.194052], [-3.623119, 37.194085], [-3.623202, 37.19413], [-3.62436, 37.196064], [-3.624439, 37.196149], [-3.62453, 37.196167], [-3.624536, 37.19619], [-3.62457, 37.196259], [-3.624605, 37.196298], [-3.624618, 37.19631], [-3.624583, 37.196363], [-3.624609, 37.196485], [-3.624742, 37.196715], {"type": "LineString"}]}, "bbox": [-3.624742, 37.18598, -3.612566, 37.196715], "type": "LineString"}], "metadata": {"attribution": "openrouteservice.org | OpenStreetMap contributors", "service": "routing", "timestamp": "1647602553377", "query": {"coordinates": [[-3.6130750179290776, 37.186441837394554], [-3.624405, 37.196839]]}, "profile": "driving-car", "format": "json"}, "engine": {"version": "6.7.0", "build_date": "2022-02-18T19:37:41Z", "graph_date": "2022-02-21T14:34:17Z"}}}
```

EJERCICIO 12: tcpdump para monitorizar el tráfico de red

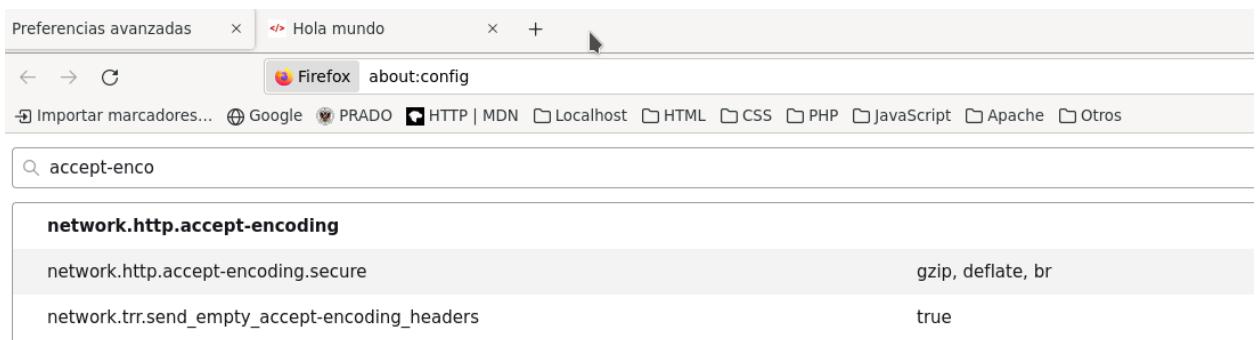
```
aulas@lubuntu:~$ sudo tcpdump -i enp0s3 -s 0 -A 'tcp dst port 80 and tcp[((tcp[12:1] & 0xf0) >> 2):4] = 0x47455420 or tcp[((tcp[12:1] & 0xf0) >> 2):4] = 0x504F5354 or tcp[((tcp[12:1] & 0xf0) >> 2):4] = 0x48545450 or tcp[((tcp[12:1] & 0xf0) >> 2):4] = 0x3C21444F and host 10.0.2.15'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
12:10:16.721358 IP lubuntu.55676 > void.ugr.es.http: Flags [P.], seq 2937617197:2937617581, ack 268800002, win 64240, length 384: HTTP: GET /tweb/hola.html HTTP/1.1
E.....@.@"*5
.....d.|.P..{....P...b...GET /tweb/hola.html HTTP/1.1
Host: void.ugr.es
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:96.0) Gecko/20100101 Firefox/96.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3
Connection: keep-alive
Upgrade-Insecure-Requests: 1
If-None-Match: "97-5c3dec54ed9f3-gzip"

12:10:16.761857 IP void.ugr.es.http > lubuntu.55676: Flags [P.], seq 1:459, ack 384, win 65535, length 458: HTTP: HTTP/1.1 200 OK
E.....@.....d
.....P.|.....HTTP/1.1 200 OK
Date: Sat, 19 Mar 2022 11:10:17 GMT
Server: Apache/2.4.46 (Ubuntu)
Last-Modified: Thu, 03 Jun 2021 16:07:23 GMT
ETag: "97-5c3dec54ed9f3"
Accept-Ranges: bytes
Content-Length: 151
Vary: Accept-Encoding
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html

<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>Hola mundo</title>
</head>
<body>
  <h1>.. Hola, mundo !</h1>
</body>
</html>
```

En la orden utilizada podemos fijarnos en algunos detalles:

- La orden utilizada es `tcpdump -i enp0s3 -s 0 -A 'tcp dst port 80 and tcp[((tcp[12:1] & 0xf0) >> 2):4] = 0x47455420 or tcp[((tcp[12:1] & 0xf0) >> 2):4] = 0x504F5354 or tcp[((tcp[12:1] & 0xf0) >> 2):4] = 0x48545450 or tcp[((tcp[12:1] & 0xf0) >> 2):4] = 0x3C21444F and host 10.0.5.12'`
- De esta orden decir que:
 - `tcp[((tcp[12:1] & 0xf0) >> 2):4] = 0x47455420`: sirve para filtrar los mensajes TCP que contengan el mensaje GET (ASCII codificado en hexadecimal).
 - `tcp[((tcp[12:1] & 0xf0) >> 2):4] = 0x504F5354`: filtrar los mensajes TCP que contengan la palabra POST.
 - `tcp[((tcp[12:1] & 0xf0) >> 2):4] = 0x48545450`: filtrar los que contengan HTTP.
 - `tcp[((tcp[12:1] & 0xf0) >> 2):4] = 0x3C21444F`: filtrar los que contengan <!DO en su inicio.
- Para conseguir capturar el mensaje sin compresión gzip he tenido que dejar vacía la configuración de firefox `accept-encoding` como se muestra a continuación:



EJERCICIO 13: ngrep para monitorizar el tráfico de red

ngrep es más sencillo de usar. Simplemente ponemos `ngrep -q 'HTTP' 'host 10.0.2.15'` y comenzamos a monitorizar el tráfico de peticiones HTTP:

```
guas@ubuntu:~$ sudo ngrep -q "HTTP" "host 10.0.2.15"
Interface: enp0s3 (10.0.2.0/255.255.255.0)
filter: ( host 10.0.2.15 ) and ((ip || ip6) || (vlan && (ip || ip6)))
match: HTTP
T 10.0.2.15:55778 -> 150.214.190.100:80 [AP] #6
    GET /tweb/hola.html HTTP/1.1..Host: void.ugr.es..User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:96.0) Gecko/20100101 Firefox/96.0..Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8..Accept-Language: es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3..Connection: keep-alive..Upgrade-Insecure-Requests: 1..If-Modified-Since: Thu, 03 Jun 2021 16:07:23 GMT..If-None-Match: "97-5c3dec54ed9f3"..Cache-Control: max-age=0...
T 150.214.190.100:80 -> 10.0.2.15:55778 [AP] #8
    HTTP/1.1 304 Not Modified..Date: Sat, 19 Mar 2022 11:39:42 GMT..Server: Apache/2.4.46 (Ubuntu)..Connection: Keep-Alive..Keep-Alive: timeout=5, max=100..ETag: "97-5c3dec54ed9f3"....
```

No he encontrado la manera de mostrar el fichero por pantalla ni en ngrep con peticiones http ni con https en ambas herramientas.