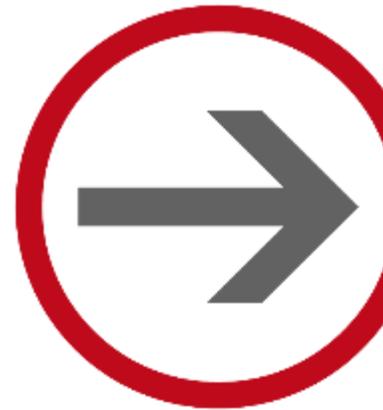




III. Explotación de servicios web

Marcelino Siles Rubia y Pablo Redondo



URL o URI? Da igual, no se lo que es

Marcelino Siles Rubia y Pablo Redondo



Universidad
Rey Juan Carlos

La estructura de una URL

`https://example.com:80/blog?search=test&sort_by=created_at#header`



Protocol



Domain



Port



Path



Query Parameters



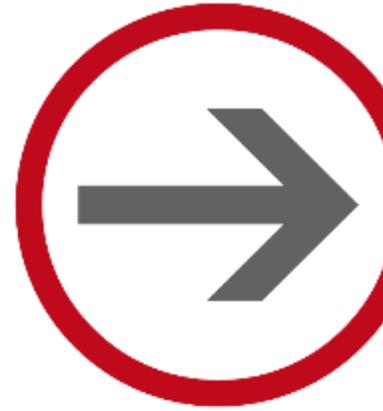
Fragment/Anchor

Lo más importante:

- Scheme → `https`
- Authority → `host:port`
- Path → `ruta del directorio`
- Parámetros → `clave:valor`

Ejemplos:

- `https://google.com/search?q=como+ganar+dinero`
- `ftp://ftp.funet.fi/pub/doc/rfc/rfc1738.txt`
- `file:///etc/passwd`
- `mailto:paco@gmail.com?subject=Notas+finales`



Arquitectura de una web

Marcelino Siles Rubia y Pablo Redondo

Archivos esenciales en una web

Los archivos esenciales son 3, un lenguaje de marcado HTML, uno de estilo CSS y otro funcional, Javascript.

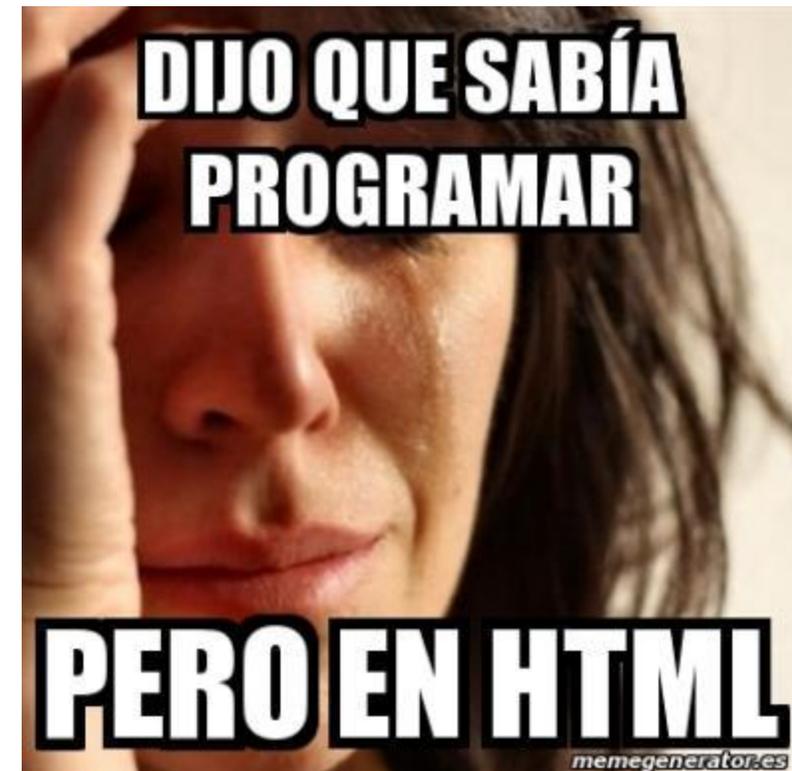


HTML

HyperText Markup Language

HTML es un lenguaje de marcado, que se forma por etiquetas y texto plano.

```
○ ○ ○  
  
<!DOCTYPE html>  
<html>  
  <head>  
    <title>Mi primera página Web</title>  
  </head>  
  
  <body>  
    <h1>  
      Mi primera página Web  
    </h1>  
    <p>This is a paragraph... </p>  
  </body>  
</html>
```



Cascading Style Sheets

Se utiliza para dar estilo a el contenido estructurado. También se puede usar con otros lenguajes como XML o SVG.

```
<style>
div {
  border: 1px solid black;
  margin-top: 100px;
  margin-bottom: 100px;
  margin-right: 150px;
  margin-left: 80px;
  background-color: lightblue;
}
</style>
```

Javascript

- Es un lenguaje de programación basado en el estándar ECMAScript de ECMA
- Las páginas web pueden incorporar interactividad con el lenguaje JavaScript
- Con JavaScript se puede modificar la página y ejecutar código cuando se interactúa con ella (a través del modelo de objetos del documento DOM)
- También se pueden hacer peticiones al servidor web en segundo plano y actualizar el contenido de la web con los resultados (AJAX)

```
const person = {
  id: 1,
  name: 'John',
  age: 23
}

// check if key exists
const hasKey = 'name' in person;

if(hasKey) {
  console.log('The key exists.');
```

```
}
else {
  console.log('The key does not exist.');
```

```
}
```

¿Pero solo existe Javascript?

Aparte de Javascript, existen muchísimos frameworks, algunos ejemplos serían PHP y Python con su módulo de Flask

PHP

```
– □ ×  
"0000" == 0 => TRUE  
"0e12" == 0 => TRUE  
"1abc" == 1 => TRUE  
"0abc" == 0 => TRUE  
"0e12345" == "0e54321" => TRUE  
"0e12345" <= "1" => TRUE
```

FLASK Y JINJA2



Y como se que framework está usando

Se suele hacer de 2 formas.



The screenshot shows the Wappalyzer interface for a website analysis. At the top left is the Wappalyzer logo and name. To the right is a link for 'Website & contact lists'. Below this, the analysis results are organized into several categories:

- CMS:** Wagtail
- JavaScript frameworks:** React 16.14.0
- Web frameworks:** Django
- Miscellaneous:** HTTP/2, webpack, Gravatar
- Programming languages:** Python
- CDN:** Cloudflare, jsDelivr
- JavaScript libraries:** jQuery 3.5.1, Modernizr 2.8.3, jQuery UI 1.12.1
- UI frameworks:** Bootstrap 4.5.2

At the bottom, there is a toggle switch for 'Create an alert for this website' and a settings gear icon.

Y como se que framework está usando

Whatweb

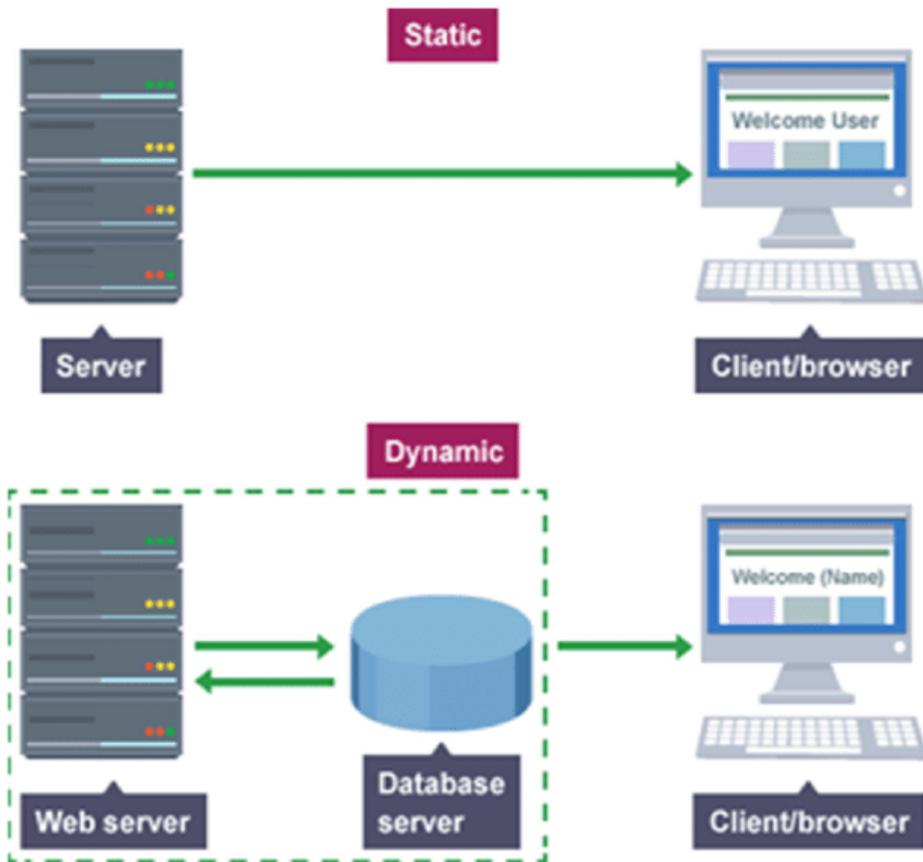
○ ○ ○

```
$ whatweb google.com
```

```
http://google.com [301 Moved Permanently] Country[UNITED STATES][US], HTTPServer[gws], IP[142.250.200.78],  
RedirectLocation[http://www.google.com/], Title[301 Moved], X-Frame-Options[SAMEORIGIN], X-XSS-Protection[0]
```

```
http://www.google.com/ [200 OK] Cookies[AEC], Country[UNITED STATES][US], HTML5, HTTPServer[gws], HttpOnly[AEC],  
IP[142.250.200.68], Script, Title[Google], X-Frame-Options[SAMEORIGIN], X-XSS-Protection[0]
```

Estático vs dinámico



Es importante saber que por regla general el usuario, no va a poder interactuar con todo el backend, si no que solo con lo que el servidor se lo permita.

Dentro de las cosas con las que puede interactuar, habrá otras que serán estáticas, y que no tendrán interacción con el cliente.

Tus nuevos amigos

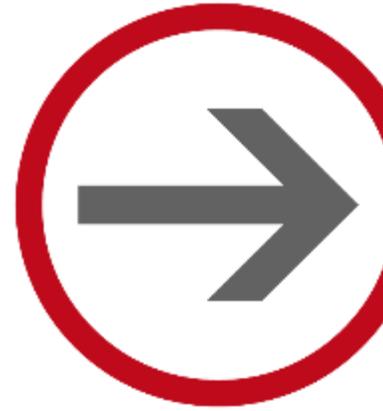
CTRL + U

Permite observar el documento (normalmente el HTML) tal cual lo recibe el servidor

F12

(o click derecho, inspeccionar elemento)

Ofrece multitud de opciones, entre ellas, la inspección del estado actual del HTML



HTTP y HTTPS

Marcelino Siles Rubia y Pablo Redondo

HTTP y HTTPS

Estos son los protocolos que hacen que la web funcione, la diferencia entre ellos es que HTTPS es HTTP con TLS, es decir cifrado.

HTTPS va a añadir los siguientes pasos al HTTP.

- Cifrar la petición con una clave simétrica
- Enviar el mensaje
- El que reciba la petición lo descifra con la misma clave simétrica

Hypertext Transfer Protocol

- La información se transmite como **texto**
- Es un protocolo **sin estado**, el servidor no tiene memoria
- Nos vamos a centrar solo en la versión **1.0/1.1**. La versión 2.0 y 3.0 son muy diferentes

```
GET /index.html HTTP/1.1
Host: google.es
Cabecera2: valor2
Cabecera3: valor3
```

Petición

```
HTTP/1.1 301 Moved Permanently
Location: http://www.google.es/
Content-Type: text/html; charset=UTF-8
Content-Length: 218
[\n\n]
<HTML><HEAD>
<TITLE>301 Moved</TITLE></HEAD><BODY>
<H1>301 Moved</H1>
The document has moved
<A HREF="http://www.google.es/">here</A>.
</BODY></HTML>
```

Respuesta

Métodos HTTP

En el ejemplo de antes se ve como realizaba un GET, pero existen más métodos HTTP.

- **POST** : Es el más utilizado junto a GET, suele servir para realizar peticiones en las que se envían datos, como podría ser un login.
- **HEAD** : Te devuelve las mismas cabeceras que si hicieras un GET pero no llega a descargar ficheros, por ejemplo si te fuera a descargar una imagen, solo te devolvería el content-length.
- **PUT** : Es similar a POST, pero es idempotente, es decir que si se realiza la misma petición varias veces, solo tendrá efecto la primera.
- **DELETE** : Borra recursos del servidor, normalmente este es un método que quieres quitar de tu web.

Si encuentras otro método y quieres investigar sobre él, existen más que los puedes investigar aquí <https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods>

Cabeceras o Headers

Hay muchas cabeceras distintas pero aquí os enumeramos las más communes.

Petición

- Referer
- Cookies
- User-agent
- Authorization
- Host

Respuesta

- Server
- Set-Cookie
- Location

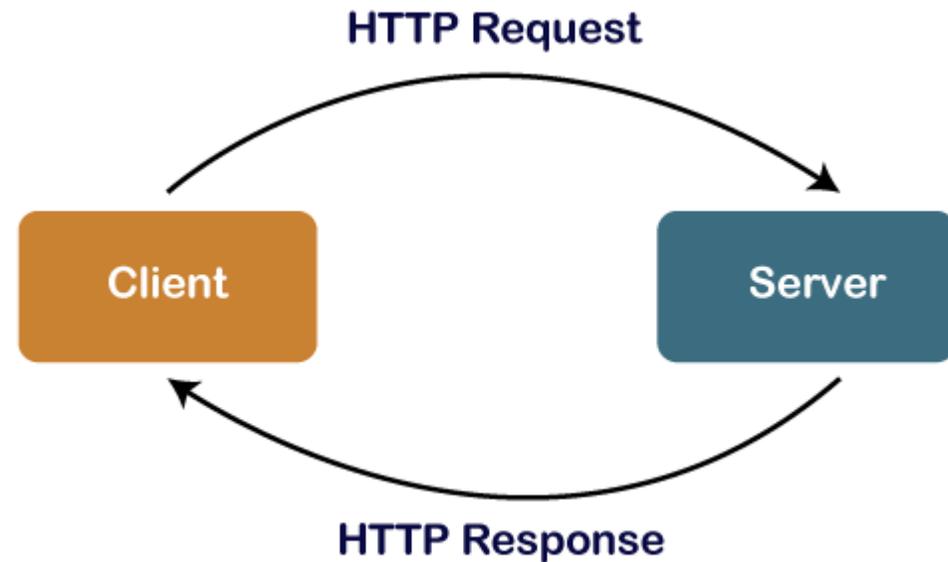
Petición

- Content-Length
- Content-Type

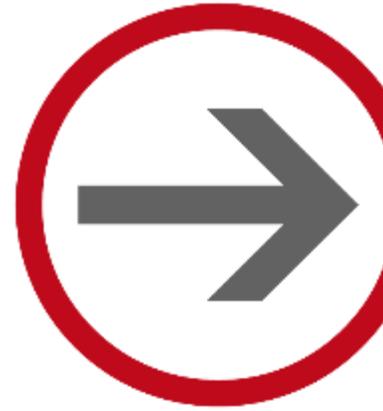
Si quieres investigar más sobre los headers aquí puedes seguir:
<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers>

Códigos de estado

¿Error 404? Ya iba tocando saber que significaba



Para investigar más sobre los código de estado: <https://developer.mozilla.org/es/docs/Web/HTTP/Status>



Fuzzing

Marcelino Siles Rubia y Pablo Redondo

¿Qué es fuzzing?

Fuzzing nos sirve para descubrir directorios, parámetros o demás campos de una página web, por fuerza bruta.

Si recordamos, lo que es el código de estado, realizando peticiones a una página web, podemos comprobar si un recurso existe o no.

Para ello existen diferentes herramientas, wfuzz, ffuf, dirbuster, gobuster, hoy os vamos a hablar de la más potente de ellas actualmente.

Ffuf es un fuzzer escrito en go.

Para instalarlo podéis hacer:

```
$ sudo apt install ffuf
```



Además vamos a utilizar los diccionarios de SecLists, que los podéis descargar así.

```
$ git clone https://github.com/danielmiessler/SecLists.git
```

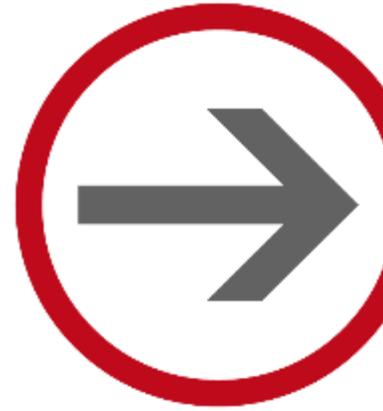
Fuzzing de directorios

Los diccionarios que yo recomiendo son los de SecLists que se encuentran en Discovery/Web-content/ en especial el directory-list-2.3-medium.txt

```
$ ffuf -c -w directorios.txt -u http://example.com/FUZZ
```

De este modo ffuf irá reemplazando la palabra FUZZ por directorios e irá probando.

```
index.html [Status: 200, Size: 16, Words: 1, Lines: 4, Duration: 64ms]
secure [Status: 301, Size: 185, Words: 6, Lines: 8, Duration: 62ms]
```



Galletas

Marcelino Siles Rubia y Pablo Redondo

Pero y como veo mis cookies

Recordamos a nuestro amigo F12

1. Inspeccionar elemento
2. Almacenamiento
3. Cookies

The screenshot shows the Chrome DevTools Storage panel. The 'Storage' tab is selected, and the 'Cookies' folder is expanded. The cookie for 'https://www.ebay.com' is selected. The table below shows the details of the cookies.

Name	Value	Domain
ak_bmsc	62B3F3547BA4FC82033D8...	.ebay.com
bm_sv	3E3CC8FE750B05C99BE45...	.ebay.com
dp1	bu1p/QEBfX0BAX19AQA**...	.ebay.com
ds2	asotr/b9TxWzzzzzzz^sotr/...	.ebay.com
ebay	%5Esbf%3D%23%5Epsi%3...	.ebay.com
nonsession	BAQAAAXrDFi8AAaAADM...	.ebay.com
npii	btguid/7fe54c381760a9b1...	.ebay.com
s	CgAD4ACBf4IYRN2ZINTRj...	.ebay.com

Y no hay una forma más fácil

Cookie editor



Cookie Editor

 Show Advanced

Search

- ▼ _ga
- ▼ _gat
- ▼ _gid
- ^ taarId

Name

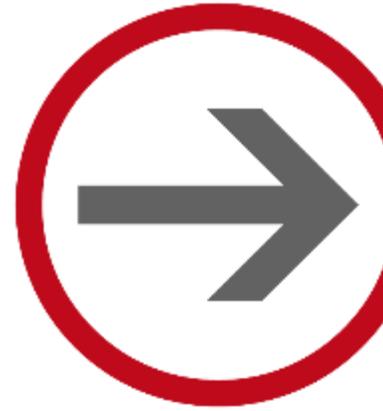
session

Value

4e627b8d78b604517a73b283f29cac7355939c16df09543dee

Show Advanced

+ [trash] [refresh] [export]



Vulnerabilidades

Marcelino Siles Rubia y Pablo Redondo

¿En lo que hemos visto ya hay vulnerabilidades?

Vamos a presentaros 2 vulnerabilidades.

- Session hijacking
- IDOR

Session hijacking

¿Cómo algo que se llama galleta puede llegar a ser malicioso?

Si la cookie no está cifrada de una forma que para el cliente no sea posible modificarla de modo que tenga sentido, puede robar la sesión de otro usuario.

```
{“usuario” : “pablito123”}
```

Suponiendo que partimos de la siguiente cookie, si la modificamos y colocamos administrador, nos estaríamos convirtiendo en el usuario administrador

```
{“usuario” : “administrador”}
```

IDOR

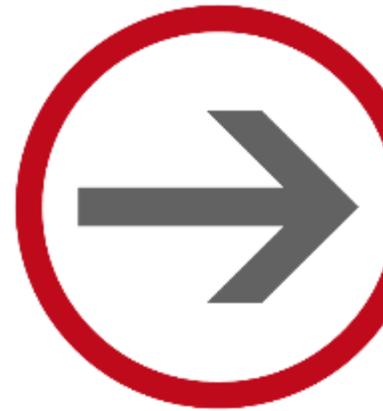
Insecure Direct Object Reference

Recordamos una parte de la URL, los parámetros, pongamos la siguiente URL como ejemplo

<http://paypal.com/cuenta?id=5>

Si podemos modificar el valor del parámetro **id** y nos renderiza la página de otro usuario habríamos conseguido entrar a la cuenta de otra persona.

<http://paypal.com/cuenta?id=4>

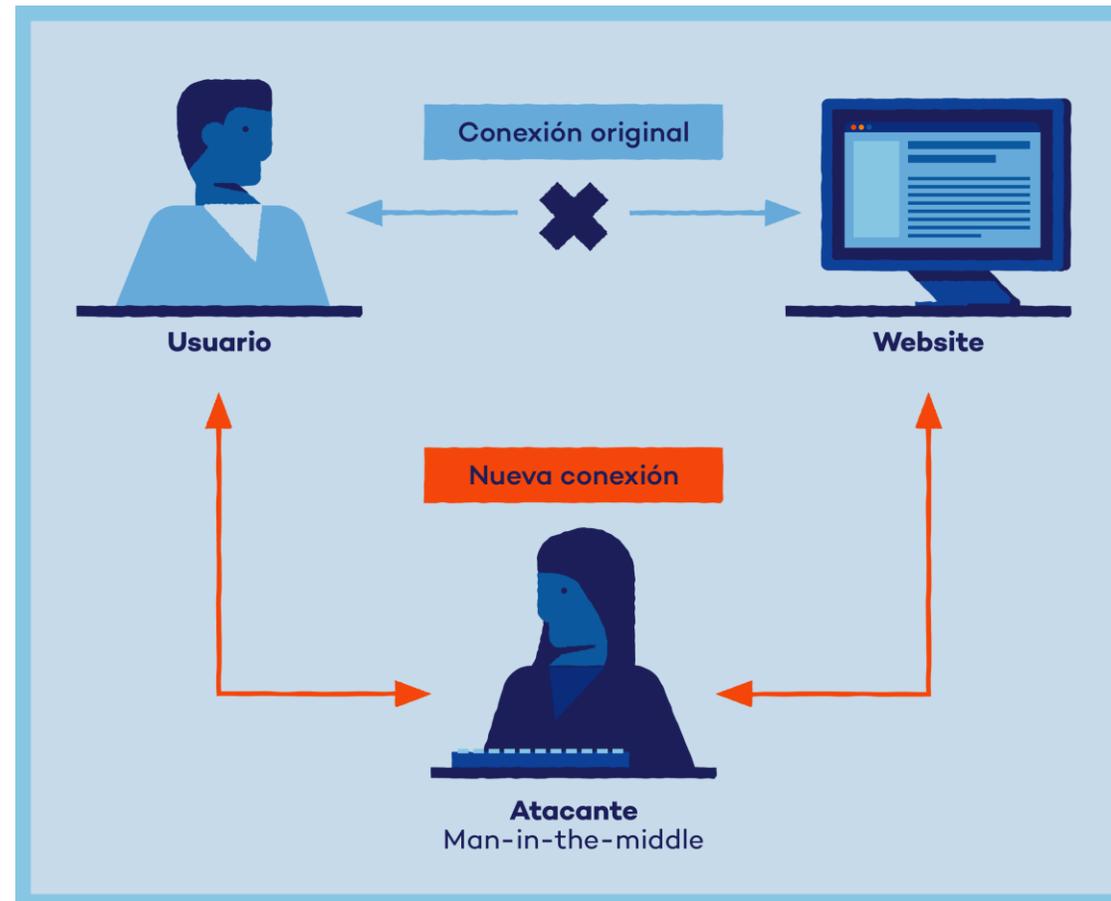


Man in the middle

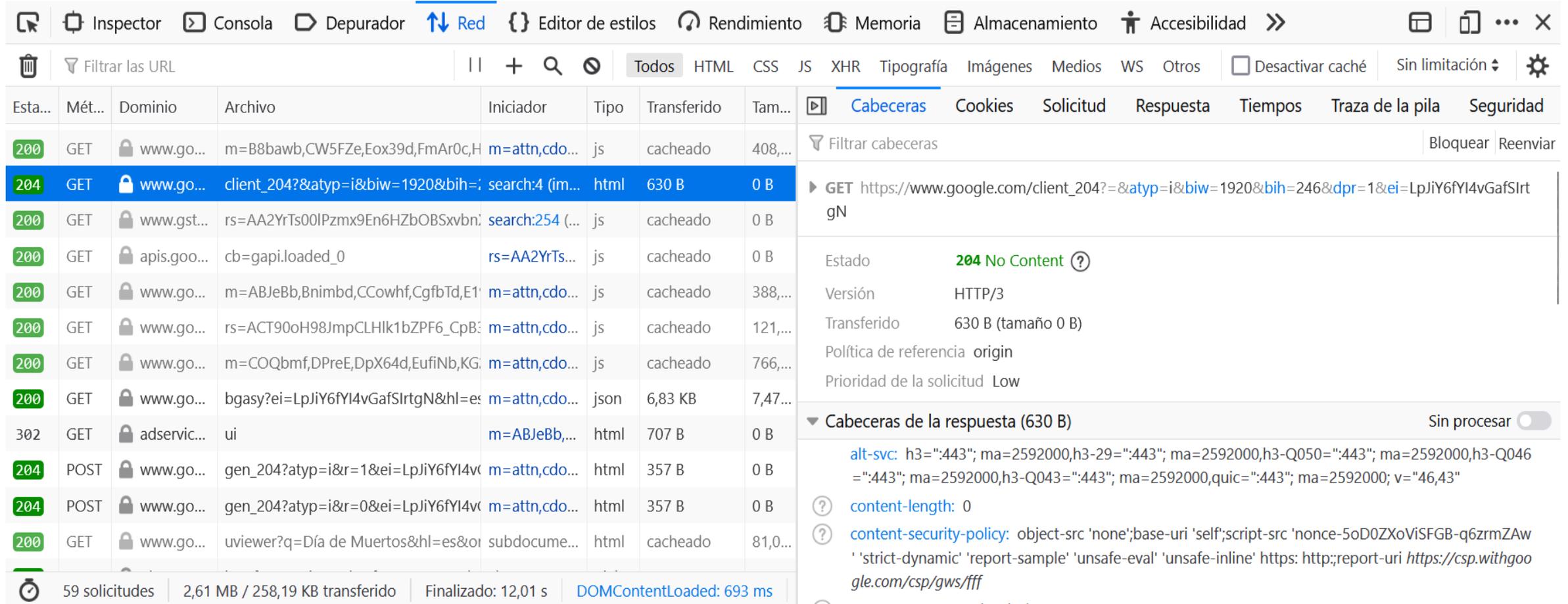
Marcelino Siles Rubia y Pablo Redondo

¿El hombre en medio?

El ataque MitM consiste en modificar la petición antes de enviarla, poniendo un punto entre medias del cliente y el servidor.



Reenviando una petición



The screenshot shows the Chrome DevTools Network tab with a list of requests. The selected request is a GET to `https://www.google.com/client_204?=&atyp=i&biw=1920&bih=246&dpr=1&ei=LpJiY6fYI4vGafSirtgN` with a status of 204 No Content. The response headers are visible on the right.

Esta...	Mét...	Dominio	Archivo	Iniciador	Tipo	Transferido	Tam...
200	GET	www.go...	m=B8bawb,CW5FZe,Eox39d,FmAr0c,H	m=attn,cdo...	js	cacheado	408,...
204	GET	www.go...	client_204?=&atyp=i&biw=1920&bih=	search:4 (im...	html	630 B	0 B
200	GET	www.gst...	rs=AA2YrTs00IPzmx9En6HZbOBSxvbn:	search:254 (...)	js	cacheado	0 B
200	GET	apis.goo...	cb=gapi.loaded_0	rs=AA2YrTs...	js	cacheado	0 B
200	GET	www.go...	m=ABJeBb,Bnimbd,CCowhf,CgfbTd,E1	m=attn,cdo...	js	cacheado	388,...
200	GET	www.go...	rs=ACT90oH98JmpCLHlk1bZPF6_CpB:	m=attn,cdo...	js	cacheado	121,...
200	GET	www.go...	m=COQbmf,DPreE,DpX64d,EufiNb,KG:	m=attn,cdo...	js	cacheado	766,...
200	GET	www.go...	bgasy?ei=LpJiY6fYI4vGafSirtgN&hl=es	m=attn,cdo...	json	6,83 KB	7,47...
302	GET	adservic...	ui	m=ABJeBb,...	html	707 B	0 B
204	POST	www.go...	gen_204?atyp=i&r=1&ei=LpJiY6fYI4v	m=attn,cdo...	html	357 B	0 B
204	POST	www.go...	gen_204?atyp=i&r=0&ei=LpJiY6fYI4v	m=attn,cdo...	html	357 B	0 B
200	GET	www.go...	uvviewer?q=Día de Muertos&hl=es&oi	subdocume...	html	cacheado	81,0...

Response Headers:

- Estado: 204 No Content
- Versión: HTTP/3
- Transferido: 630 B (tamaño 0 B)
- Política de referencia: origin
- Prioridad de la solicitud: Low

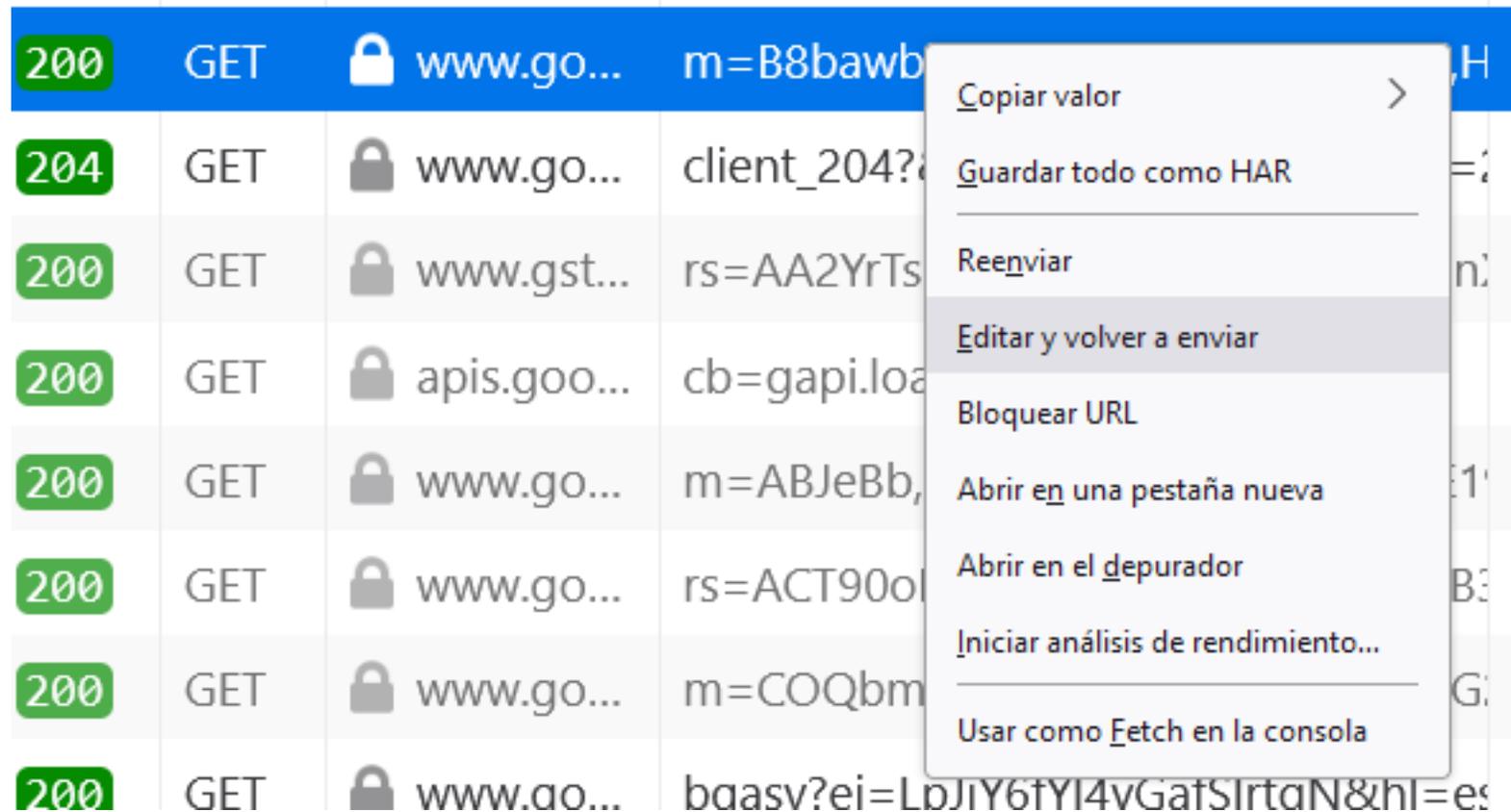
Response Headers (630 B):

- `alt-svc`: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000,h3-Q050=":443"; ma=2592000,h3-Q046=":443"; ma=2592000,h3-Q043=":443"; ma=2592000,quic=":443"; ma=2592000; v="46,43"
- `content-length`: 0
- `content-security-policy`: object-src 'none';base-uri 'self';script-src 'nonce-5oD0ZXoViSFGB-q6zrmZAw' 'strict-dynamic' 'report-sample' 'unsafe-eval' 'unsafe-inline' https: http;report-uri https://csp.withgoogle.com/csp/gws/fff

Summary: 59 solicitudes | 2,61 MB / 258,19 KB transferido | Finalizado: 12,01 s | DOMContentLoaded: 693 ms

Reenviando una petición

Para reenviar una petición hay que darle click derecho, editar y reenviar



200	GET	www.go...	m=B8bawb...	...
204	GET	www.go...	client_204?	...
200	GET	www.gst...	rs=AA2YrTs...	...
200	GET	apis.goo...	cb=gapi loa...	...
200	GET	www.go...	m=ABJeBb,	...
200	GET	www.go...	rs=ACT90o...	...
200	GET	www.go...	m=COQbm...	...
200	GET	www.ao...	baasv?ei=L...	...

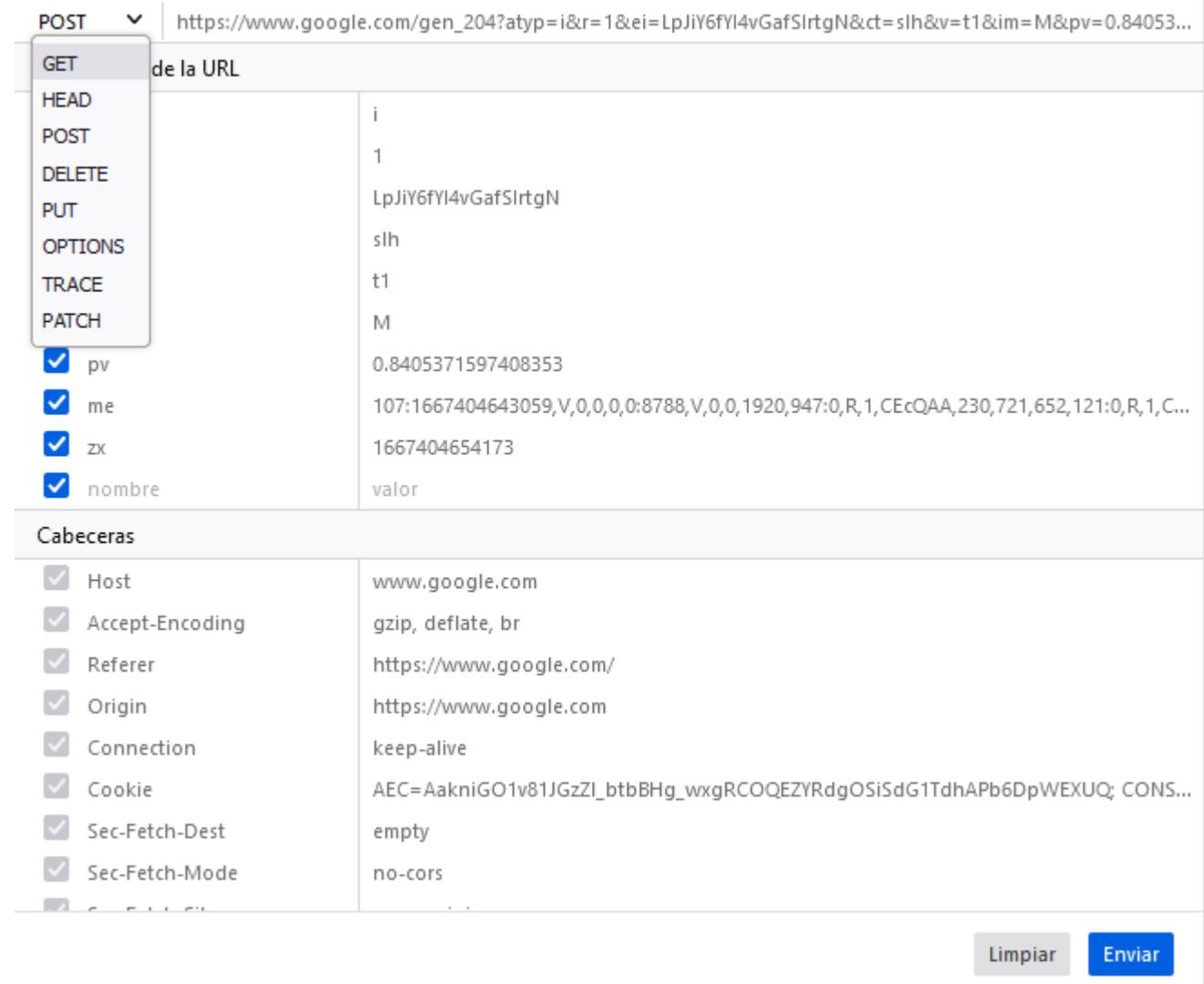
- Copiar valor >
- Guardar todo como HAR
- Reenviar
- Editar y volver a enviar
- Bloquear URL
- Abrir en una pestaña nueva
- Abrir en el depurador
- Iniciar análisis de rendimiento...
- Usar como Fetch en la consola

Reenviando una petición

Aquí dentro podemos hacer:

- Modificar el método.
- Modificar cabeceras o añadir nuevas.
- Cambiar los valores de los parámetros.

Finalmente dándole a enviar se enviaría la petición modificada.



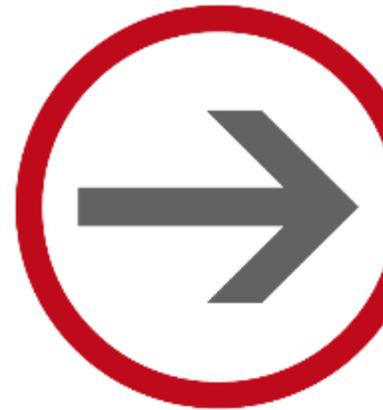
POST GET HEAD POST DELETE PUT OPTIONS TRACE PATCH

de la URL

i	
1	
LpJiY6fYI4vGafSlrtgN	
sIh	
t1	
M	
<input checked="" type="checkbox"/> pv	0.8405371597408353
<input checked="" type="checkbox"/> me	107:1667404643059,V,0,0,0,0:8788,V,0,0,1920,947:0,R,1,CEcQAA,230,721,652,121:0,R,1,C...
<input checked="" type="checkbox"/> zx	1667404654173
<input checked="" type="checkbox"/> nombre	valor

Cabeceras

<input checked="" type="checkbox"/> Host	www.google.com
<input checked="" type="checkbox"/> Accept-Encoding	gzip, deflate, br
<input checked="" type="checkbox"/> Referer	https://www.google.com/
<input checked="" type="checkbox"/> Origin	https://www.google.com
<input checked="" type="checkbox"/> Connection	keep-alive
<input checked="" type="checkbox"/> Cookie	AEC=AakniGO1v81JGzZI_btBHg_wxgRCOQEZYRdgOSiSdG1TdhAPb6DpWEXUQ; CONS...
<input checked="" type="checkbox"/> Sec-Fetch-Dest	empty
<input checked="" type="checkbox"/> Sec-Fetch-Mode	no-cors



Burp Suite

Marcelino Siles Rubia y Pablo Redondo

Burp Suite

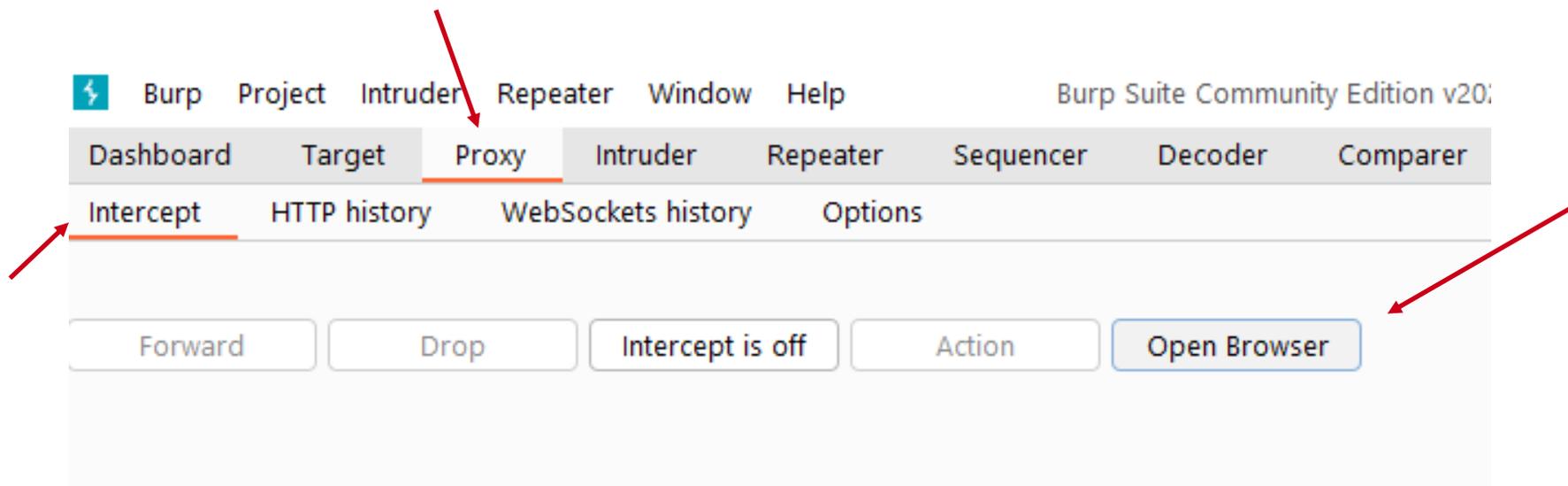
Burp Suite nos va a ayudar con el ataque de man in the middle, ya que nos va a simplificar el hecho de coger peticiones y modificarlas.

- Proxy que redirige el tráfico
- Pausa la petición
- Se modifica
- Se envía la petición



El proxy

- Para utilizar Burp Suite, por defecto estará escuchando en el localhost por el puerto 8080, por lo que tenemos que redirigir el tráfico a nuestro puerto 8080 interno.
- Por simplicidad podemos usar el navegador integrado que tiene Burp Suite, que ya redirige el tráfico automáticamente.



Intercept

Proxy → Intercept

Forward

Drop

Intercept is on

Las distintas utilidades de intercept son las siguientes:

- Forward : Envía la petición
- Drop : Suelta la petición y no la envía
- Intercept is on/off : Bloquea / Libera el tráfico

Petición GET

Una petición GET de ejemplo, como podemos observar los parámetros se pasan por la URL

```
1 GET /search?q=exploit+para+joomla HTTP/1.1
2 Host: github.com
3 Cookie: user_session=DhjtVjwCewClpgRSBRryV3fd2JBCAwPhjufUWzQt2RVCKeQP
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:106.0)
  Gecko/20100101 Firefox/106.0
5 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,ima
  ge/webp,*/*;q=0.8
6 Accept-Language: es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3
7 Accept-Encoding: gzip, deflate
8 Referer: https://github.com/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
```

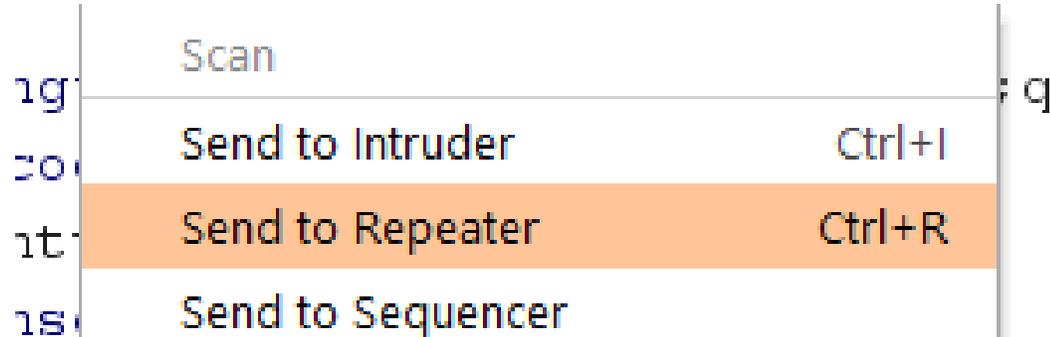
Petición POST

Una petición POST, los parámetros se pasan en la URL como objetos

```
POST /search HTTP/1.1
Host: github.com
Cookie: user_session=DhjtVjwCewC1pgRSBRryV3fd2JBCAwPhjufUWzQt2RVCKeQP
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:106.0) Gecko/20100101
Firefox/106.0
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
Accept-Encoding: gzip, deflate
Referer: https://github.com/
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Te: trailers
Content-Type: application/x-www-form-urlencoded
Content-Length: 21

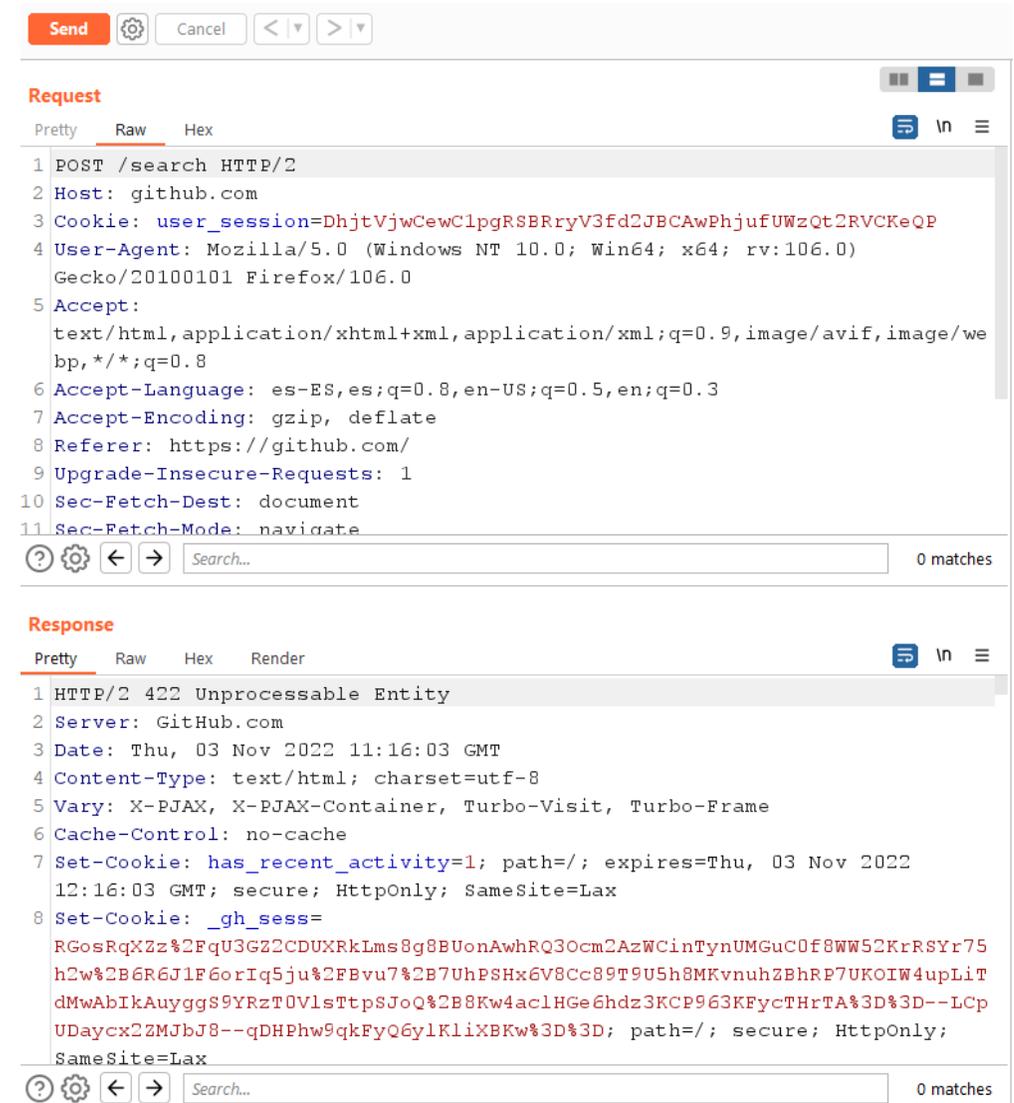
search=exploit+para+joomla
```

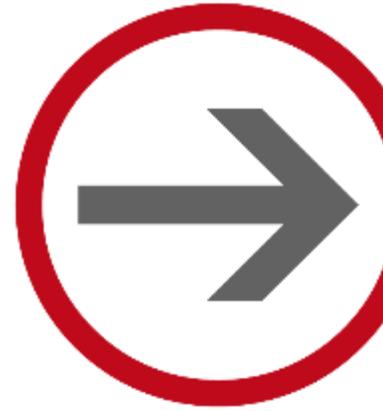
Repeater



- Click derecho + Send to Repeater
- Ctrl + R

En el Repeater te vas a poder guardar una petición, para poder hacerle cambios, y después de darle a send, observar la respuesta, es útil cuando estamos probando distintos input en una misma petición.





SQL Injection

Marcelino Siles Rubia y Pablo Redondo



Universidad
Rey Juan Carlos

¿Qué es SQL?

Structured Query Language

SQL es un lenguaje para bases de datos donde su principal estructura son las “query” o peticiones. Estas pueden servir tanto para insertar datos, como para recuperar datos de las tablas.



```
INSERT INTO tabla (columna1, columna2, columna3) VALUES (valor1, valor2, valor3);
```

```
SELECT * FROM tabla WHERE columna = valor;
```

Ejemplo de queries de MySQL

¿Pero y cuando inyecto?

Es importante recalcar que hay diferentes tipos de bases de datos, y que no todas tienen la misma sintáxis, por simplicidad vamos a hablar solo de MySQL

Una inyección de MySQL consiste en modificar una query, de tal modo que cuando se realice devuelva valores para la que no estaba intencionada.

Primero, vamos a comentaros que son las operaciones lógicas OR y AND, ya que son fundamentales para esto.

AND & OR

Las operaciones AND & OR son operaciones lógicas, que actúan de la siguiente forma.

AND Truth Table

Inputs		Output
A	B	$Y = A.B$
0	0	0
0	1	0
1	0	0
1	1	1

OR Truth Table

Inputs		Output
A	B	$Y = A+B$
0	0	0
0	1	1
1	0	1
1	1	1

Comentarios en SQL

Hay varias formas de comentar en MySQL, las más comunes son las siguientes

- Doble guión y un espacio “– “

Nótese la importancia del espacio para comentar, por ello lo que se suele hacer es poner “– -”

- Almohadilla “#”

La primera inyección SQL

Dada la siguiente query SQL

```
SELECT username FROM usuarios WHERE username = '$user' AND password = '$passwd'
```

Si el usuario introduce como usuario **administrador'-- -**

Escapará el contexto de las comillas, y podrá comentar el resto.

```
SELECT username FROM usuarios WHERE username = 'administrador'-- -'AND PASSWORD = '$passwd'
```

SQL Injection

Y si no conocemos el nombre de usuario, como podemos hacerlo. Aquí entran en juego las operaciones lógicas.

Si el usuario inyecta **admin' OR 1=1-- -**

```
SELECT username FROM usuarios WHERE username = 'admin' OR 1=1-- -'AND PASSWORD = '$passwd'
```

Aunque no exista un usuario que se llame admin, la query será válida, porque 1 es igual a 1, y solo 1 de las 2 partes tiene que ser válida.

```
s WHERE username = 'admin' OR 1=1-- -'AND PASSWORD = '$passwd'
```



SQL Injection UNION BASED

UNION SELECT

- Las SQL Injection Union Based, nos van a servir en la mayoría de casos, para exfiltrar datos de la base de datos, para ello utilizamos UNION SELECT, el cual concatena la misma cantidad de valores que los que se solicitan en la primera query, y del mismo tipo.
- Es decir si originalmente, el primer SELECT, escoge username e id, con UNION SELECT podrás recoger 2 parámetros más, uno de tipo string y otro de tipo int.
- Además no tiene porque ser de la misma tabla.

Pasos para lograr una UNION BASED

1º Encontrar el número de columnas, que la query original está recuperando.

Para esto vamos a usar ORDER BY N, que iremos incrementando el N hasta que nos de error, ya que no podrán ordenar las columnas, ya que le has especificado un número mayor del que hay.



```
SELECT a, b FROM table1 ORDER BY 2-- - OK!
```

```
SELECT a, b FROM table1 ORDER BY 3-- - ERROR!
```

Tendrías que inyectar algo parecido a **'ORDER BY 3-- -**

Pasos para lograr una UNION BASED

2º Enumerar cuales de los campos se reflejan y son string

Para ello utilizaremos NULL, el cual siempre será valido en cualquier tipo de campo.

OK!

```
' UNION SELECT 'a', NULL-- -
```

ERROR!

```
' UNION SELECT NULL, 'a'-- -
```

```
SELECT username, id FROM users WHERE user='$user';
```

Pasos para lograr una UNION BASED

3º Enumerar la information_schema

Sacar los nombres de las tablas

```
UNION SELECT table_name, NULL FROM information_schema.tables-- -
```

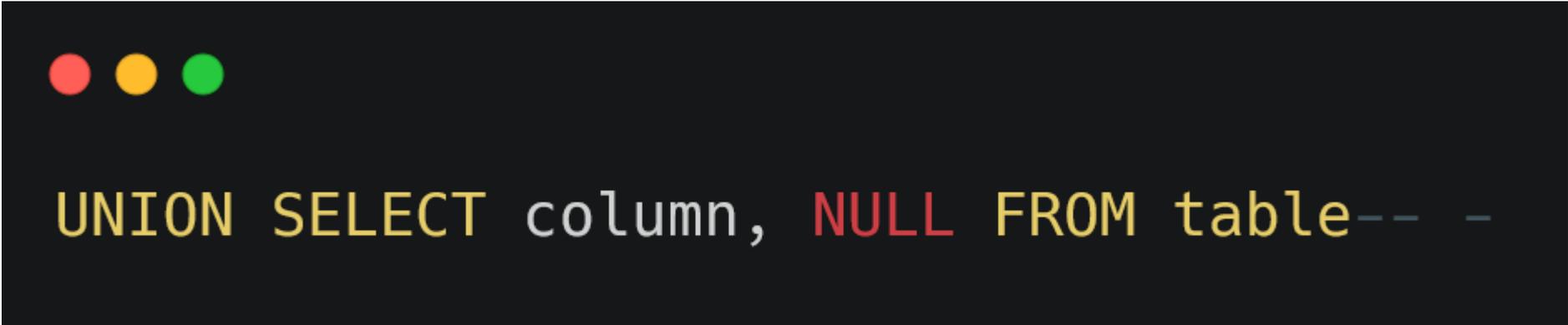
Sacar las columnas de una tabla

```
SELECT group_concat(column_name) FROM information_schema.columns WHERE table_name = 'Tabla'
```

Pasos para lograr una UNION BASED

4º Recuperar datos conociendo el nombre de la tabla y las columna

Conociendo el nombre de la tabla, y de las columnas que quremos ya podemos extraer los datos.



```
UNION SELECT column, NULL FROM table-- -
```

La query sería **'UNION SELECT column, NULL FROM table-- -**



Universidad
Rey Juan Carlos