

# Informe Laboratorio 3

## Sección 1

Alumno Diego Banda Gálvez  
e-mail: diego.banda@mail\_udp.cl

Octubre de 2025

## Índice

<b>1. Descripción de actividades</b>	<b>2</b>
<b>2. Desarrollo de actividades según criterio de rúbrica</b>	<b>2</b>
2.1. Identifica el algoritmo de hash utilizado al momento de registrarse en el sitio	2
2.2. Identifica el algoritmo de hash utilizado al momento de iniciar sesión . . . . .	5
2.3. Genera el hash de la contraseña desde la consola del navegador . . . . .	6
2.4. Intercepta el tráfico login con BurpSuite . . . . .	7
2.5. Realiza el intento de login por medio del hash . . . . .	9
2.6. Identifica las políticas de privacidad o seguridad . . . . .	11
2.7. Comente 4 conclusiones sobre la seguridad del sitio escogido . . . . .	12

## 1. Descripción de actividades

Su objetivo será auditar la implementación de algoritmos hash aplicados a contraseñas en páginas web desde el lado del cliente, así como evaluar la efectividad de estas medidas contra ataques de tipo Pass the Hash (PtH). Para llevar a cabo esta auditoría, deberá registrarse en un sitio web y crear una cuenta, ingresando una contraseña específica para realizar las pruebas.

Al concluir la tarea, es importante que modifique su contraseña por una diferente para garantizar su seguridad.

Dado que la cantidad de sitios chilenos que utilizan hash es limitada, se permite realizar esta tarea en cualquier sitio web a nivel mundial. En este sentido, realice las siguientes actividades:

- Identificación del algoritmo de hash utilizado para las contraseñas al momento del registro en el sitio.
- Identificación del algoritmo de hash utilizado para las contraseñas al momento de iniciar sesión.
- Generación del hash de la contraseña desde la consola del navegador, partiendo de la contraseña en texto plano.
- Interceptación del tráfico de login utilizando BurpSuite desde su equipo.
- Realización de un intento de login modificando la contraseña por una incorrecta haciendo uso del hash obtenido en el punto anterior. Puede interceptar el tráfico y modificar el hash por el correcto o hacer uso del servicio repeater de BurpSuite.
- Descripción de las políticas de privacidad o seguridad relacionadas con las contraseñas, incluyendo un enlace a las mismas.
- Cuatro conclusiones sobre la seguridad o vulnerabilidad de la implementación observada.

## 2. Desarrollo de actividades según criterio de rúbrica

Para el laboratorio en esta ocasión se utilizó la página [mmo-champion.com](http://mmo-champion.com), la cual a través de búsqueda y prueba se consideró óptima para desarrollar las actividades de este laboratorio.

### 2.1. Identifica el algoritmo de hash utilizado al momento de registrarse en el sitio

Primero se comienza realizando el registro en la página con un correo temporal, las credenciales son las siguientes:

## 2 DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

- **User Name:** tareacripto
- **Password:** tareacripto123
- **Email Address:** sahiw26669@gta5hx.com

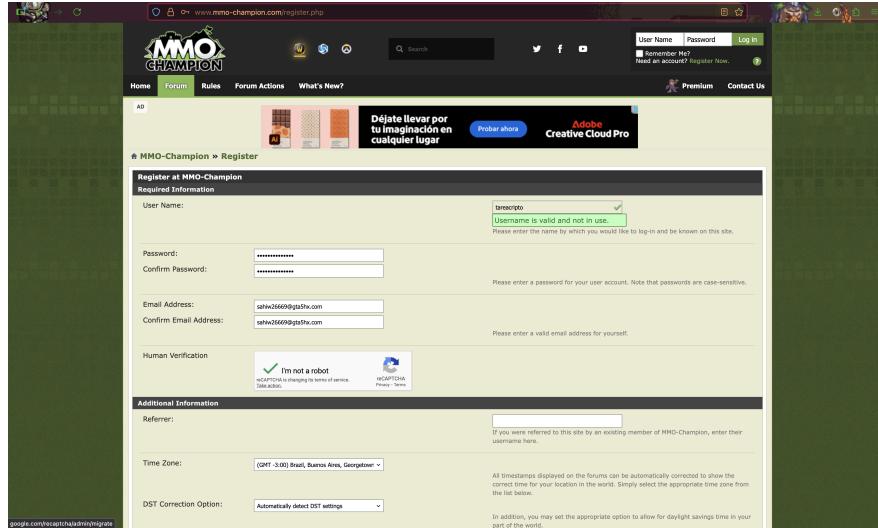


Figura 1: Registro en MMO-Champion

Status	Method	Domain	File	Initiator	Type	Transferred	Size
200	POST	www.mmo-champion...	register.php?do=addmember	document	html	12.76 kB	38.45 kB
200	GET	dt.adsafeprotected...	dtAdvertinityId=218410&adId=32168bd8-aadc-a7cf-e90-8ea77e2df9	img	gif	246 B	43 B
200	GET	www.mmo-champion...	v3.css?20240327v3	stylesheet	css	cached	-1 B
200	GET	www.mmo-champion...	css.php?styleId=9&langId=1&d=1740165569&t=lt&sheet=bbc.cs	stylesheet	css	cached	-1 B
200	GET	ajax.googleapis.com	yuloader-dom-event.js	script	js	cached	61.62 kB
200	GET	www.mmo-champion...	vbulletin-core.js?v=425	script	js	cached	51.93 kB
200	GET	ajax.googleapis.com	jquery.min.js	script	js	cached	95.99 kB
200	GET	ajax.googleapis.com	jquery-ui.min.js	script	js	cached	237.1 kB
304	GET	www.wowdb.com	tt.js	script	js	cached	18.26 kB
200	GET	static.wowdb.com	mv.js	script	js	4.21 kB	8.56 kB
200	GET	static.mmo-champion...	mmogallery.js?20210308	script	js	cached	11.91 kB
200	GET	www.mmo-champion...	vbulletin_mds.js?v=425	script	js	cached	5.46 kB
200	GET	media.mmo-champio...	adsense.js	script	js	cached	21 B

Figura 2: Formulario enviado para registro

## 2 DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

---

A través de la pestaña *Network* de las herramientas de desarrollador de los navegadores, realizamos un análisis de las comunicaciones y procesos realizados a la hora de confirmar el registro, identificando lo siguiente: La comunicación *html* seleccionada del tipo *POST* es aquella que hace envío de los datos puestos en el formulario de registro, el formulario enviado al *back-end* es el siguiente:

Listing 1: Formulario de registro enviado al *back-end*

```
1 username= "tareacripto"
2 password= ""
3 passwordconfirm= ""
4 email= "sahiw26669@gta5hx.com"
5 emailconfirm= "sahiw26669@gta5hx.com"
6 humanverify[hash]= "e10fa32becd3df6b91387690ef11e1f5"
7 g-recaptcha-response= "0cAFcWeA5wDdUzi1XVuGKV0aI2R8tg090MiYMTpG7
FkSc4ux64e5xcIOjeN..." // Recortado ya que no es de utilidad
8 user4112024791= ""
9 referrername= ""
10 timezoneoffset= "-3"
11 dst= "2"
12 options[adminemail]= "1"
13 agree= "1"
14 s= ""
15 securitytoken= "guest"
16 do= "addmember"
17 url= "https://www.mmo-champion.com/content/"
18 password_md5= "b95f072cbad1494685833f32f63ce61a"
19 passwordconfirm_md5= "b95f072cbad1494685833f32f63ce61a"
20 day= ""
21 month= ""
22 year= ""
```

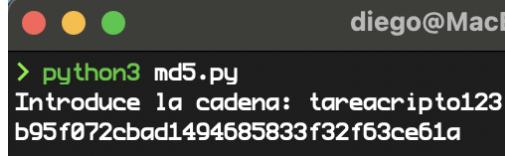
Se pueden observar claros indicios del manejo de la contraseña por parte del sitio web, por ejemplo el segundo y tercer parámetro enviado es '*password*' y '*passwordconfirm*' respectivamente, sin embargo están vacíos, no se envía lo ingresado en texto plano. Después de varias líneas de información, que no es útil para este laboratorio, podemos llegar a los parámetros '*password\_md5*' y '*passwordconfirm\_md5*', con esto podemos identificar que se está *hasheando* con *MD5* aparte del nombre que tiene, su longitud es de 32 caracteres hexadecimal, y se creó el siguiente código de *python* con ayuda de *chatGPT* para *hashear* en *MD5* y comparar el resultado, así confirmar que nos encontramos ante tal tipo de *hash*.

Listing 2: Código de python cadena de texto a md5 (github)

```
1 import hashlib
2
3 s = input("Introduce la cadena: ")
4 md5_hex = hashlib.md5(s.encode('utf-8')).hexdigest()
5 print(md5_hex)
```

## 2 DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

---



```
diego@MacBook-Pro:~/Desktop$ python3 md5.py
Introduce la cadena: tareacripto123
b95f072cbad1494685833f32f63ce61a
```

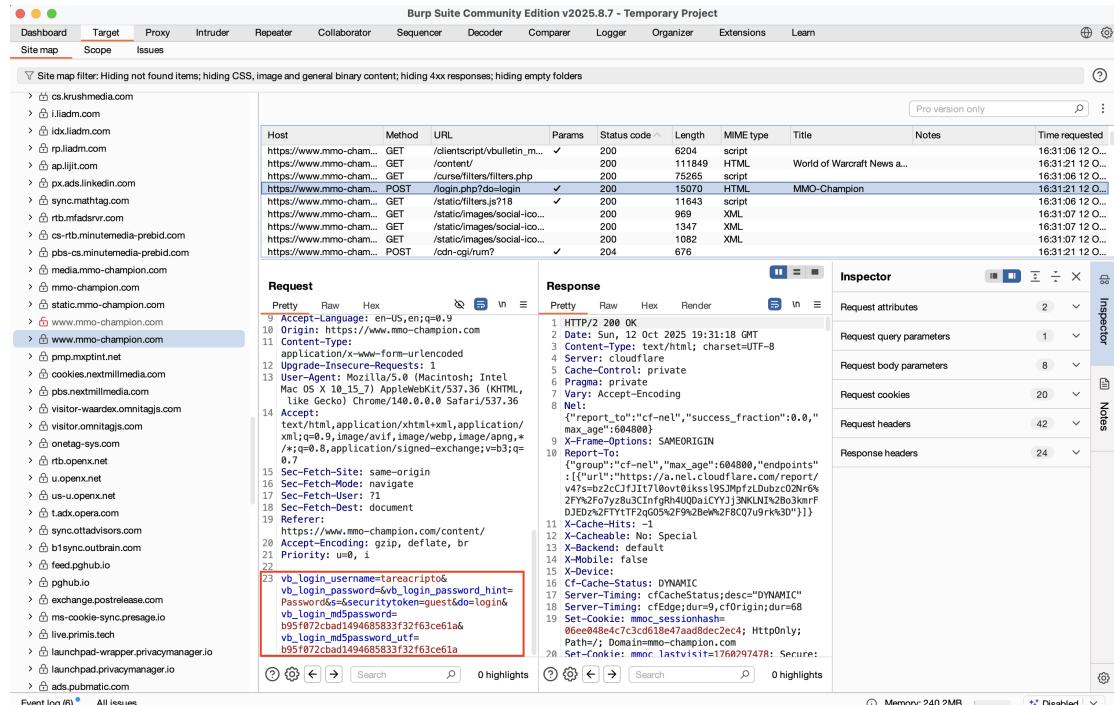
Figura 3: Resultado *md5* en python

- Cadena entregada por la página: 'b95f072cbad1494685833f32f63ce61a'.
- Cadena entregada por *python*: 'b95f072cbad1494685833f32f63ce61a'.

Como se observa, ambas cadenas son iguales, por lo tanto, gracias a estas comprobaciones finales podemos confirmar al 100 % que el *hash* utilizado por la página es *MD5*.

### 2.2. Identifica el algoritmo de hash utilizado al momento de iniciar sesión

En esta ocasión se realizó la captura con *burpsuite*, ya que así es más fácil analizar los paquetes, utilizando un navegador común, estos se borraban por una redirección de la página, además en esta aplicación se separan en grupos automáticamente, siendo mejor para la identificación de la petición de inicio de sesión.



Host	Method	URL	Params	Status code	Length	MIME type	Title	Notes	Time requested
https://www.mmo-champ...	GET	/clientscript/vbulletin_m...		200	6204	script			16:31:06 12 O...
https://www.mmo-champ...	GET	/content/		200	111849	HTML	World of Warcraft News a...		16:31:21 12 O...
https://www.mmo-champ...	GET	/curve/filter/filter.php		200	75265	script			16:31:06 12 O...
<b>https://www.mmo-champ...</b>	<b>POST</b>	<b>/login.php?do=login</b>		<b>200</b>	<b>15070</b>	<b>HTML</b>	<b>MMO-Champion</b>		<b>16:31:21 12 O...</b>
https://www.mmo-champ...	GET	/static/filter.js?18		200	11643	script			16:31:06 12 O...
https://www.mmo-champ...	GET	/static/images/social-icon...		200	940	XML			16:31:07 12 O...
https://www.mmo-champ...	GET	/static/images/social-icon...		200	1347	XML			16:31:07 12 O...
https://www.mmo-champ...	GET	/static/images/social-icon...		200	1082	XML			16:31:07 12 O...
https://www.mmo-champ...	POST	/cdn-cgi/rum?		204	676				16:31:21 12 O...

Figura 4: Formulario enviado para inicio de sesión

## 2 DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

La comunicación *html* del tipo *POST* hizo envío al *back-end* lo siguiente:

Listing 3: Formulario de inicio de sesión enviado al *back-end*

```

1 vb_login_username= "tareacripto"
2 vb_login_password= ""
3 vb_login_password_hint= "Password"
4 s= ""
5 securitytoken= "guest"
6 do= "login"
7 vb_login_md5password= "b95f072cbad1494685833f32f63ce61a"
8 vb_login_md5password_utf= "b95f072cbad1494685833f32f63ce61a"

```

Igual que en el paso anterior, vemos de qué manera maneja las contraseñas el sitio, no envía tal parámetro importante en texto plano, sino que *hasheada* con *MD5* nuevamente, esto se puede identificar porque literalmente lo dice en el parámetro enviado, nuevamente es una cadena de 32 caracteres hexadecimal y es la misma cadena enviada en el paso anterior (ya que es la misma contraseña), por lo tanto si se compara con el *hasheo* hecho con *python* si se puede confirmar que es del tipo mencionado anteriormente.

### 2.3. Genera el hash de la contraseña desde la consola del navegador

The screenshot shows the Burp Suite interface with the following details:

- Request:**

```
1 GET /clientscript/vbulletin_md5.js?v=425
HTTP/2.0
2 Host: www.mmo-champion.com
3 Cookie: ...; JSESSIONID=Gh1.2.1494685833f32f63ce61a; ...; _lrc_fpi=c9c8be18e2dd...; _lrc_meta=47b22...; _lrc_retry_request=true; _lrc_env_src_attrs=false; _scor_uid=d83ba8940f874075908a5c2d5d3158f; _lfpwi=ZTVMTU0ZGYM1I1Yy00Z2K1TkwZD1Y1N1ZD14NMWmM2h11y5LTU3D...; panormald_expiry=176032667999; __cc_id=ec382bf78a3a34c2921dfed001f1b384; __gads=ID=fda7ae021378633f4;t=1760296269;RT=1760296269;S=ALNI_MZ1d1qd-8jdfCWg1YflJg4950UA;__gpi=UID=000129d05327c77:T=1760296269:RT=1760296269:S=ALNI_NZ141sHrn04XGtwLvsQXUMT-I=0;__eo1=ID=6ae7baa451d8c3b0:T=1760296269:RT=1760296269:S=AA-AfjzdrE457mg0aJ-pxJElYMU;__ga_GCBRNJWVH=GS2.1.1760296266$01g1$t1760296292$934$10$0
4 Sec-Ch-Ua-Platform: "macOS"
```
- Response:**

```
21 var hexcase=0;
var b64pad="";
var chrsz=8;
function hex_md5(A){
    return bin2hex(core_md5(str2binl(A,A.length
        *chrsz)));
}
function b64_md5(A){
    return bin2b64(core_md5(str2binl(A,A.length
        *chrsz)));
}
function str_md5(A){
    return bin2str(core_md5(str2binl(A,A.length
        *chrsz)));
}
function hex_hmac_md5(A,B){
    return bin2hex(core_hmac_md5(A,B));
}
function b64_hmac_md5(A,B){
    return bin2b64(core_hmac_md5(A,B));
}
function str_hmac_md5(A,B){
    return bin2str(core_hmac_md5(A,B));
}
function core_md5(K,F){
    K[F>5]=128<-(F%32);
    K[((F>64)>>9)<>4]+14=F;
    var J=1732584193;
    ...
```
- Inspector:** Shows the request and response attributes, query parameters, cookies, headers, and response headers.
- Notes:** Shows the time requested for each item.

Figura 5: Llamado a funciones

## 2 DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

A través de los mismos paquetes capturados en el punto anterior se identificó esta función llamada *hex\_md5* la cual recibe como parámetro una cadena de texto y retorna el resultado de su *hash* de *md5*, si introducimos la contraseña utilizada (*tareacripto123*) obtenemos el mismo *hash* mostrado anteriormente.

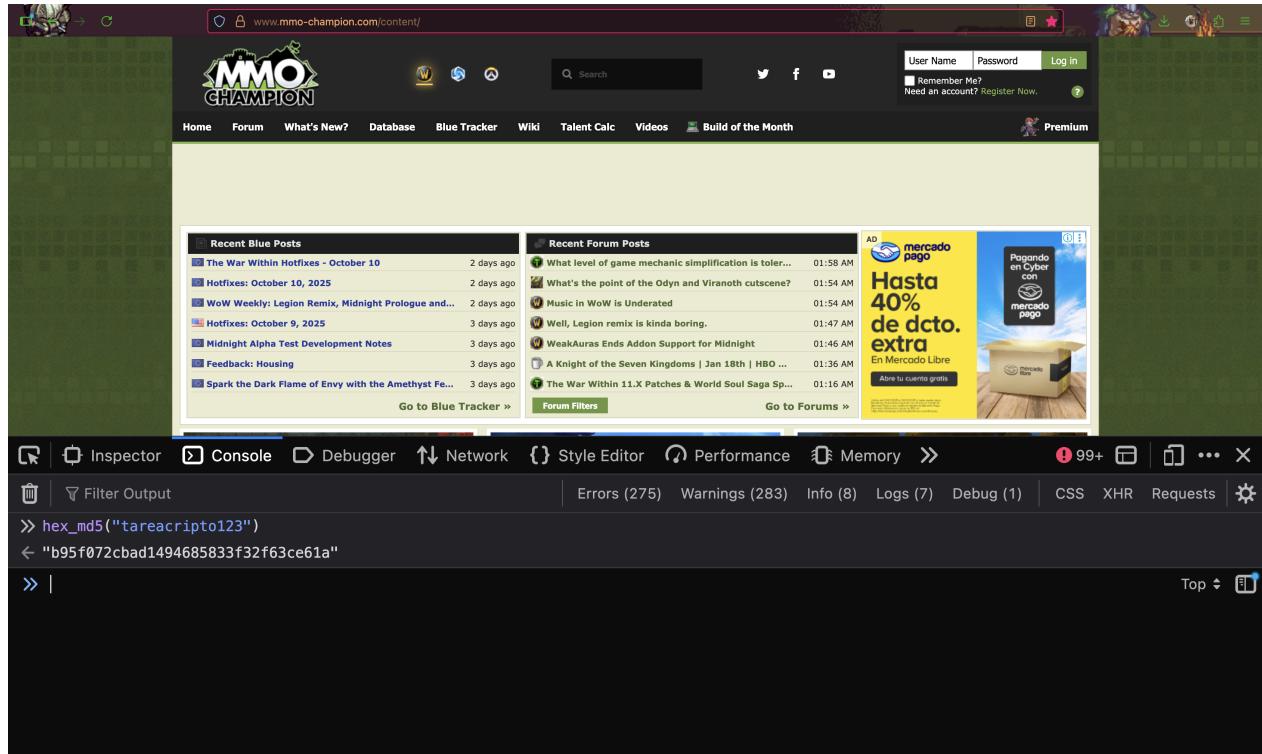


Figura 6: Llamado a función *hex\_md5(tareacripto123)*

Claramente se obtiene la misma cadena 'b95f072cbad1494685833f32f63ce61a' vista anteriormente.

### 2.4. Intercepta el tráfico login con BurpSuite

Se realizó la captura de los paquetes a la hora de entrar a la página y hacer un inicio de sesión correcto con burpsuite, a través de su pestaña '*Proxy*', la captura se ve de la siguiente manera en la aplicación:

## 2 DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

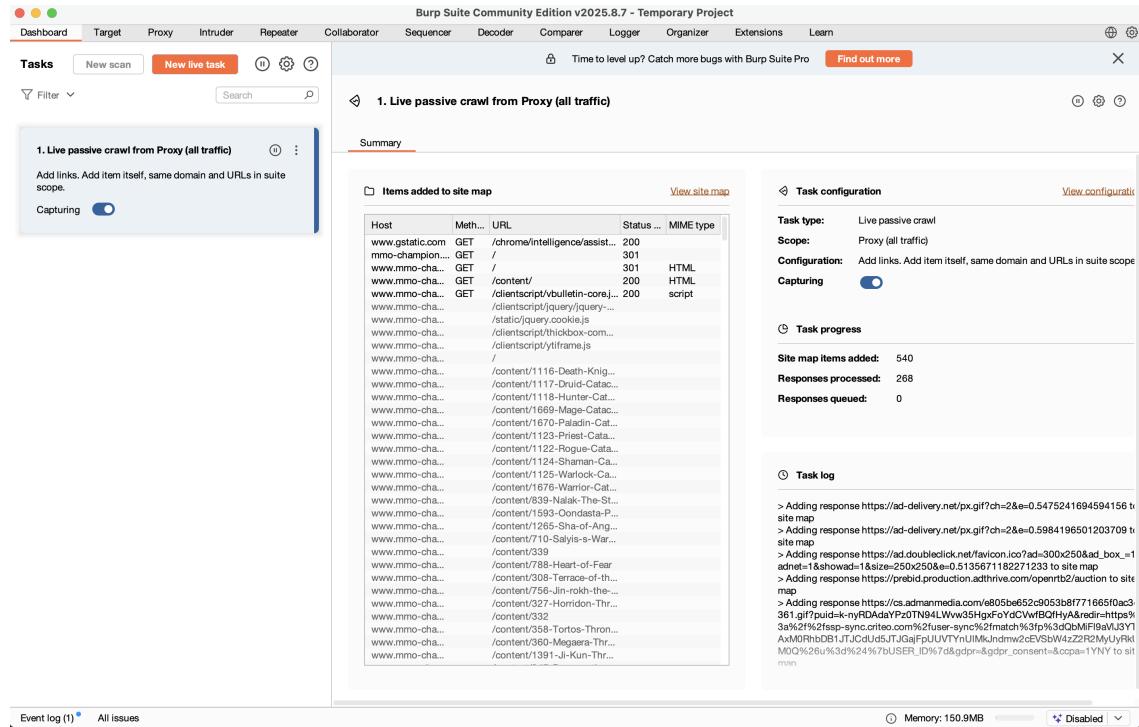


Figura 7: Intersección de tráfico de la página web a través de brupsuite (*Dashboard*)

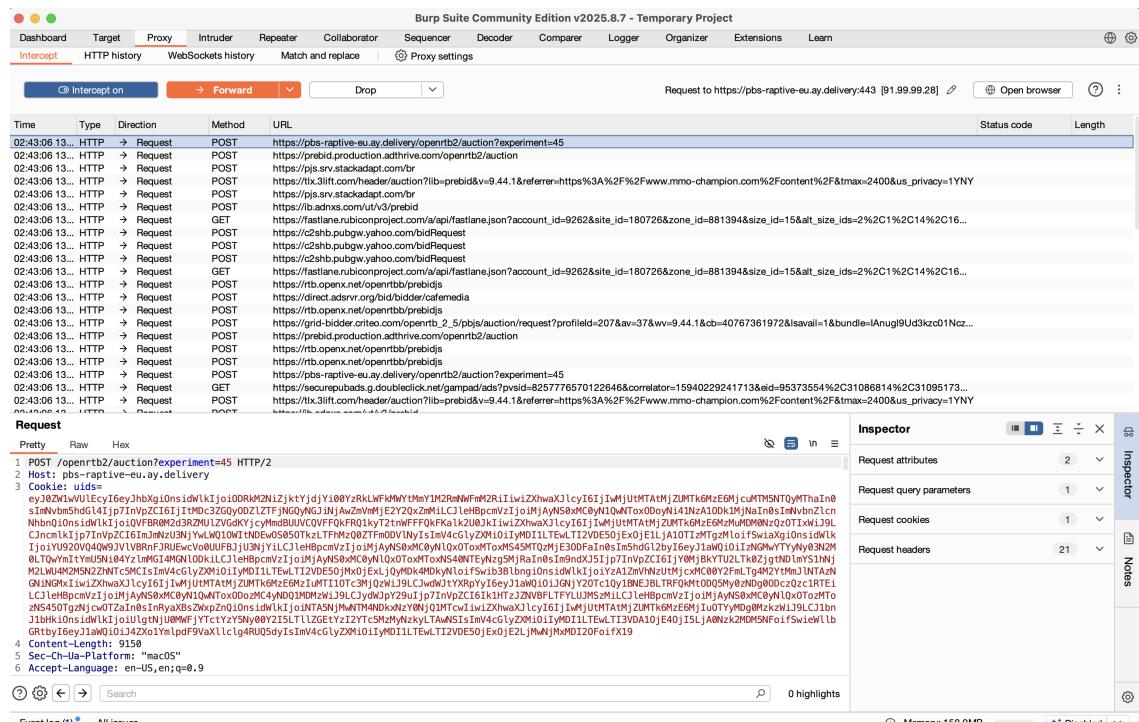


Figura 8: Intersección de tráfico de la página web a través de brupsuite (*Proxy*)

## 2 DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

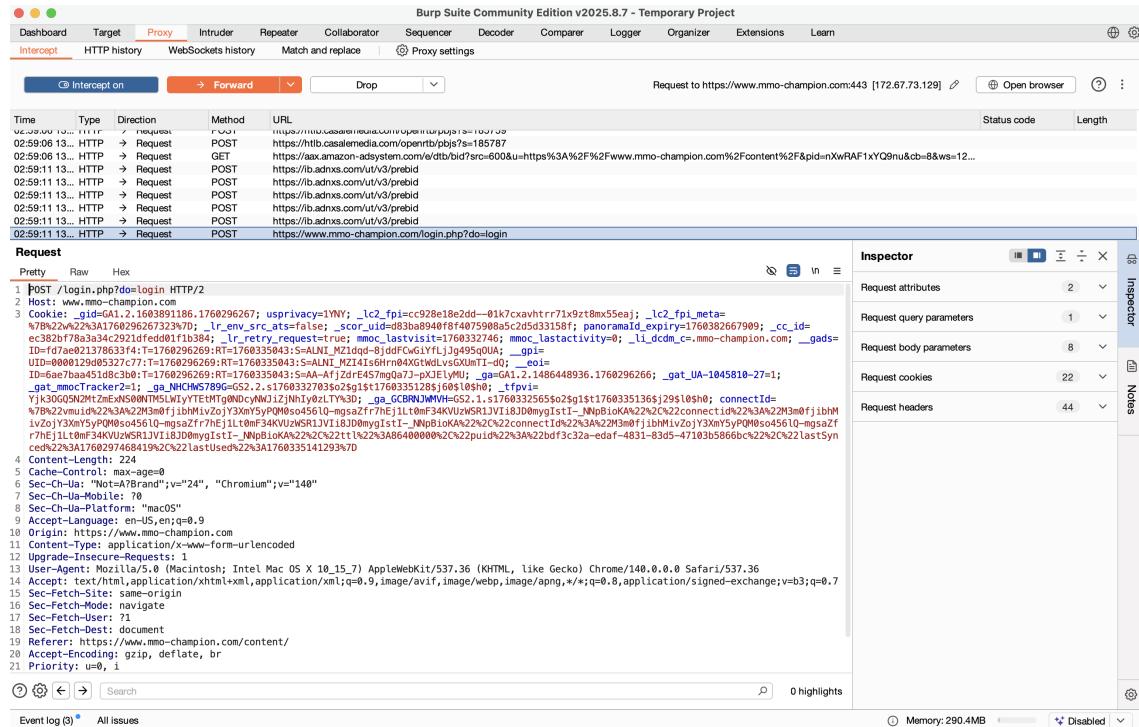


Figura 9: Intersección de tráfico de la página web a través de brupsuite (*Proxy* inicio de sesión)

Navegando a través de los paquetes obtenidos gracias a esta intersección se pueden identificar varios llamados a la *api* realizados desde el *front-end*, lo enviado en los formularios, datos, scripts utilizados, entre otros. También llama mucho la atención la cantidad de paquetes exteriores a la propia página web que se capturan, siendo estos muchos de publicidad, re-direcciones, entre otros, pero principalmente los mencionados, esto puede ser utilizado por los desarrolladores de la página para ganar dinero y sacar un beneficio de levantar tal servicio, sin embargo esto entorpece un poco el análisis, ya que se debe explorar aún más entre todos estos dominios y llamados que se realizan, para así identificar correctamente lo que si sirve y será utilizado.

Se entiende esta herramienta para poder interceptar el tráfico como una muy poderosa, siendo capaz de modificar los paquetes a gusto del usuario antes de enviarlos, y según como funcione la página web y su *back-end*, se podrían causar problemas de seguridad.

### 2.5. Realiza el intento de login por medio del hash

Sin tener la contraseña, solo el *hash* de aquella contraseña, igualmente se puede realizar un inicio de sesión correcto ingresando una contraseña cualquiera, esto a través de interceptar el tráfico generado por la página web, modificar el formulario enviado con el *hash* correcto, el *back-end* identificará tal *hash* como correcto, por lo tanto dará la autorización para acceder.

El procedimiento es el siguiente: Ingresar a la página web, arriba a la derecha se encuentra

## 2 DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

el formulario de inicio de sesión, se llena con el nombre de usuario correspondiente, sin embargo una contraseña aleatoria distinta de la correspondiente, antes de hacer click en 'Log in' se activa la opción de interceptar en *burpsuite*, esto capturará el paquete de la petición, entregando la capacidad de modificarlo a nuestro gusto antes de enviar tal petición, aquí es donde se reemplaza el *hash* de la contraseña incorrecta con el de la contraseña correcta, lo siguiente es enviar el paquete y esperar la respuesta del servidor.

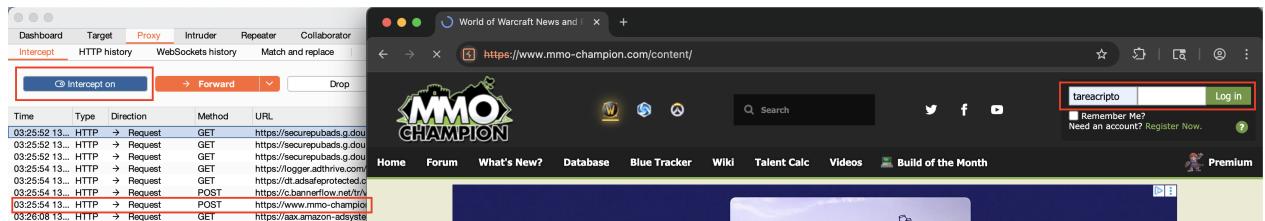


Figura 10: Captura de petición *POST* con formulario de inicio de sesión

El paquete a modificar es aquel de *login*, ya que este envía el formulario con lo ingresado (*username* y *password hasheada*). Se identifica gracias a que este hace un llamado a una url con nombre representativo, también porque este contiene los parámetros necesarios para llevar a cabo tal acción, en este caso es el *username* 'tareascripto' y el *hash* de la contraseña incorrecta 'hola'.

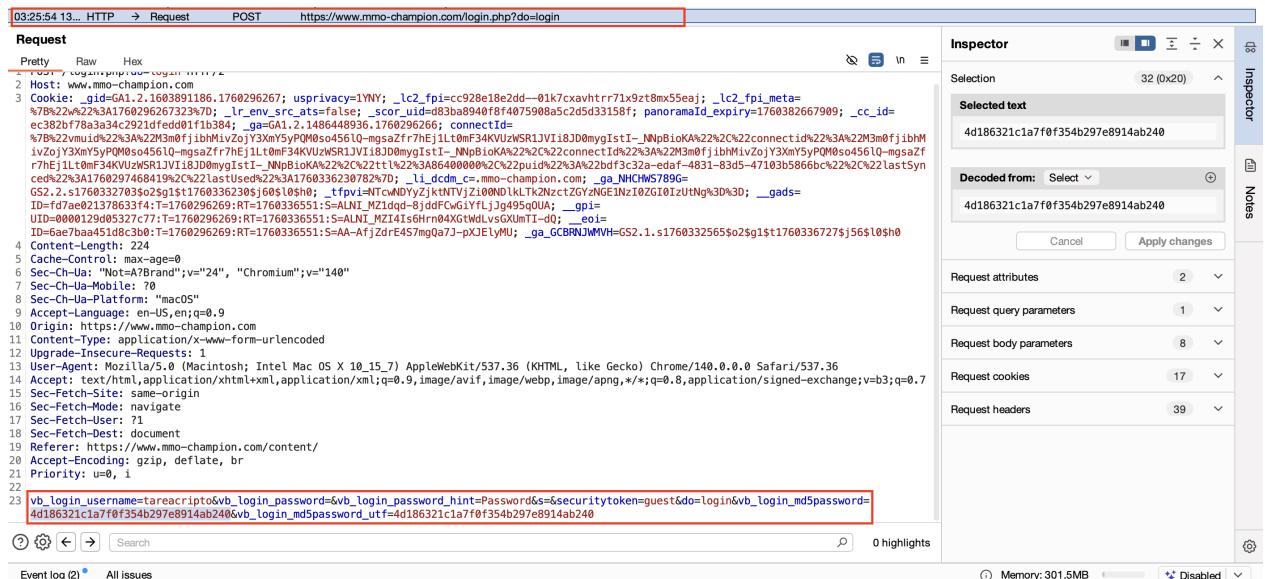
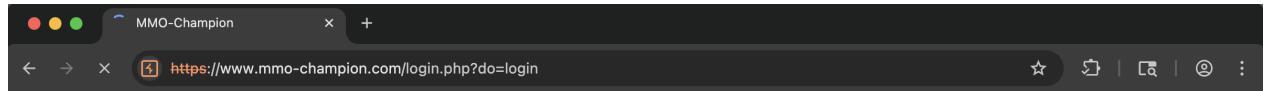


Figura 11: Paquete a modificar y parámetros a modificar

Como se aprecia en la imagen, el *hash* es el de la contraseña incorrecta, este se modifica, reemplazandolo por el correcto (visto en pasos anteriores 'b95f072cbad1494685833f32f63ce61a'), luego de esto se hace clic en la funcionalidad *Forward* de *burpsuite* para enviar tal petición,

## 2 DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

también podría ser *Forward all* para enviar esta y todas las posibles siguientes que se intercepten. En las siguientes imágenes se puede observar como el inicio de sesión se realizó correctamente:



### Redirecting...

Thank you for logging in, tareacripto.

[Click here if your browser does not automatically redirect you.](#)

[Terms of Content Use](#)

Figura 12: Re-direccionamiento realizado por la página cuando se inicia sesión

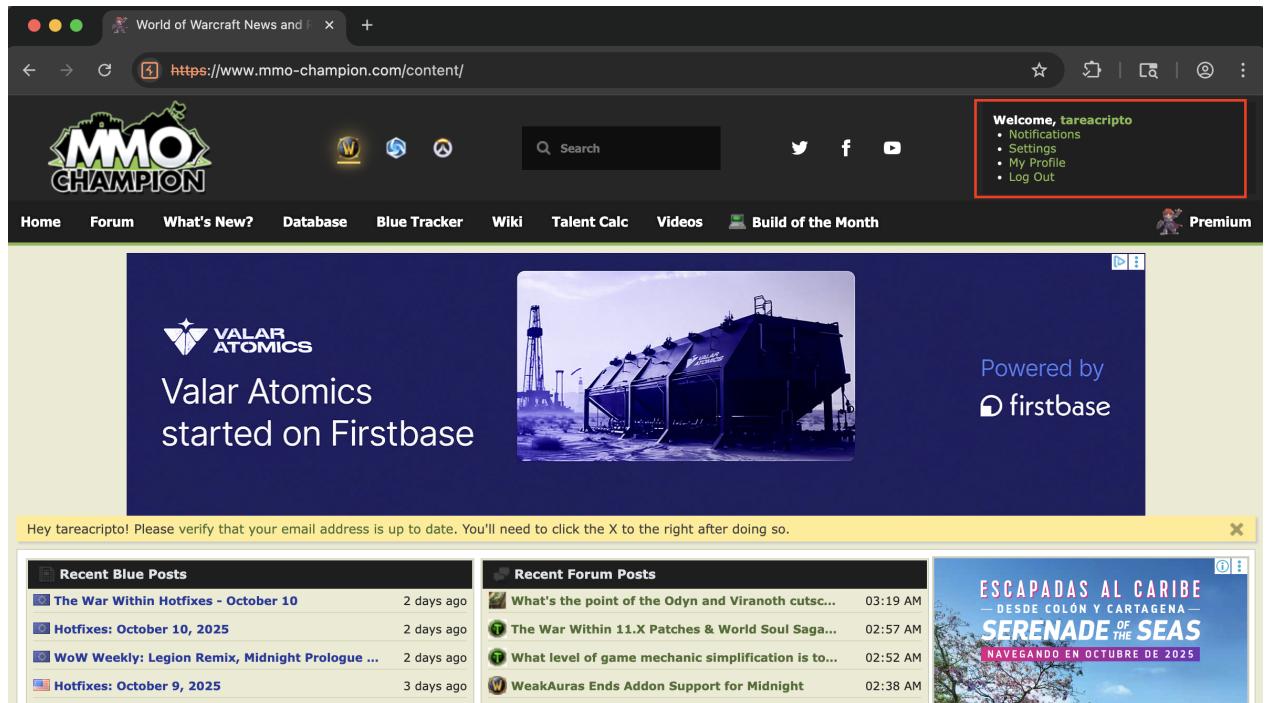


Figura 13: Sesión iniciada correctamente

### 2.6. Identifica las políticas de privacidad o seguridad

En cuanto a la página seleccionada, no existe mucha información en ella sobre la seguridad aplicada, lo más cercano a ello, fueron estas dos:

- [magicfind.us/privacy/](http://magicfind.us/privacy/)
- [www.mmo-champion.com/faq.php](http://www.mmo-champion.com/faq.php)

En la primera página se habla sobre que cumplen con las leyes de la Unión Europea, Estados Unidos entre otros, lo cual representa distintas cosas, como por ejemplo, estar obligados a no almacenar las contraseñas en texto plano, no utilizar cifrado posiblemente reversible para *hashing*, contar con autenticador multi-factor (MFA), entre otros. Sin embargo, las leyes cambian constantemente a través del tiempo, sobretodo en esta era digital que avanza a pasos gigantes hoy en día, por lo tanto, puede que las medidas de seguridad que se estén aplicando no estén actualizadas a las últimas leyes y existan falencias en tal aspecto.

Por otro lado, desde este apartado de preguntas frecuentes, lo único que se logra obtener es que, efectivamente se está haciendo uso de algoritmos de *hash* no reversibles para almacenar las contraseñas, ya que hablan de la encriptación y la imposibilidad de enviar de vuelta la contraseña en caso de perderla.

### 2.7. Comente 4 conclusiones sobre la seguridad del sitio escogido

1. **Uso de *hash*:** El uso de *hash* en el *front-end* antes de hacer el envío del formulario con la contraseña en texto plano es algo crucial para la seguridad en las redes, ya que de lo contrario, alguien podría fácilmente interceptar el tráfico y la contraseña quedaría expuesta, esta es una práctica que todo sitio debería realizar.
2. **Uso de *md5* para *hash*:** Como se explica en el punto anterior, el uso de *hash* para las contraseñas es crucial, sin embargo, existen problemas con el sitio web. Primero, es muy fácil identificar qué tipo de *hash* está siendo utilizado, lo cual le juega en contra, sobre todo por el tipo de *hash* que es, *md5* es un tipo de *hash* que dejó de ser confiable aproximadamente el 2004, y gracias a esto puede ser fácilmente vulnerado para generar colisiones de *hash* con las contraseñas.
3. **Solo uso de contraseña para *hash*:** El utilizar solo la contraseña ingresada para realizar el *hash* puede implicar en más colisiones y con tener una fuente de datos proveniente de la página, podríamos predecir u obtener tales contraseñas solo con ingresar la original, es por ello que siempre se recomienda el uso de algún *salt*, idealmente único para cada usuario (por ejemplo, rut, *username* si este es único, etc.). Con esto se reducirían bastante los posibles problemas de colisión entre personas con contraseñas iguales o muy similares.
4. **Posibilidad de activar el (MFA):** Una medida muy segura y que actualmente es muy robusta ante ataques es el multi-factor authentication, ya que requiere de una autorización por parte del usuario en otro medio o similar, este punto es bueno, ya que la página si cuenta con la posibilidad de usar tal tecnología, se reducen muchísimo los problemas con posibles ataques.

## 2 DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

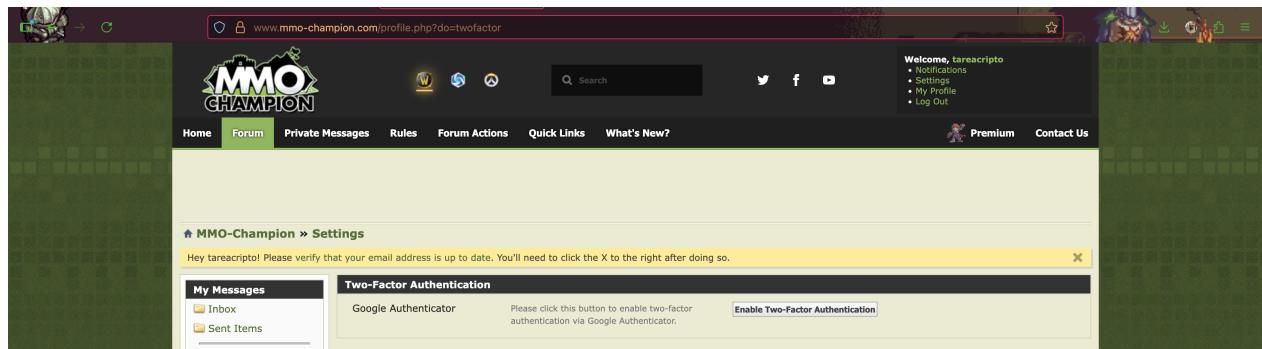


Figura 14: MFA activación