

# Informe Laboratorio 3

## Sección 2

Cristóbal Barra  
cristobal.barra1@mail.udp.cl

Octubre de 2024

## 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 . . . . .	8
2.6. Identifica las políticas de privacidad o seguridad . . . . .	10
2.7. Demuestra 4 conclusiones sobre la seguridad . . . . .	11

## 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 una contraseña incorrecta por el hash obtenido en el punto anterior.
- 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

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

Para la realización de este laboratorio, se hizo uso de una de las páginas web recomendadas por el profesor, la cual es **LinuxQuestions.org**. Dentro de esta web se hará la identificación del hash que se utiliza para guardar las contraseñas de manera “segura”.

Figura 1: Formulario de registro.

Para identificar el algoritmo de hash utilizado para registrarse dentro del sitio, será necesario entrar en el modo inspección de elemento, dentro del cual será posible apreciar el documento HTML. Bajando hasta la sección de registro de usuario y yendo hasta la subsección para ingresar la contraseña y la confirmación de ésta misma, se puede observar que en los campos ocultos se encuentran unos parámetros llamados **password\_md5** y **passwordconfirm\_md5**, lo que ya nos dice que las contraseñas generadas serán hasheadas con el algoritmo MD5.

```
<form action="register.php?do=addmember" name="register" method="post" onsubmit="return verify_passwords(password, passwordconfirm);">
  <input type="hidden" name="s" value="">
  <input type="hidden" name="securitytoken" value="guest">
  <input type="hidden" name="do" value="addmember">
  <input type="hidden" name="url" value="https://www.linuxquestions.org/">
  <input type="hidden" name="agree" value="1">
  <input type="hidden" name="password_md5">
  <input type="hidden" name="passwordconfirm_md5">
  <input type="hidden" name="day" value="0">
  <input type="hidden" name="month" value="0">
  <input type="hidden" name="year" value="0">
```

Figura 2: Inspección de elemento.

Para generar el hash a través del registro, debemos rellenar los campos de la figura 1, para ello usaremos el usuario *CriptoLab* y la contraseña *1234*, junto con el correo temporal creado en **yopmail.com**, *criptolab@yopmail.com*.

Luego del registro, se generará un paquete con el método POST, el cual contiene las credenciales con las que nos registramos anteriormente, tal como se aprecia en la figura 3 plasmada a continuación.

Estado	Método	Dominio	Archivo	Iniciador	Tipo	Transferido	Tamaño
200	POST	www.linuxquestions.org	register.php?do=addmember	document	html	7,91 KB	23,35 KB
	POST	rum-http-intake.logs...	pubdfe3d3f97883d30bebd7d86fb4401f1d7ddsource=browser&ddtags=sd_	beacon	json	NS_ERROR_FAILURE	0 B
	POST	rum-http-intake.logs...	pubdfe3d3f97883d30bebd7d86fb4401f1d7ddsource=browser&ddtags=sd_	datadog-rum.js:1 (beacon)	json	NS_ERROR_FAILURE	0 B
200	GET	www.google.com	anchor?ar=1&k=6LcDp_45AAAAAB59mRc8MYxkZUC_n9wdOaQMwjmW8co:	subdocument	html	32,04 KB	49,75 KB
200	GET	www.gstatic.com	recaptcha__en.js	script	js	cacheado	0 B

Figura 3: Inspección del paquete POST generado al momento del registro.

Dentro del paquete, nos dirigimos a la subsección Request situada a la derecha, la cual nos dejará ver el payload del paquete. Dentro de éste, se aprecian los campos de usuario, email y contraseña, aunque este último se encuentra vacío con el fin de proteger las contraseñas, sin embargo, más arriba se encuentra la contraseña hasheada con MD5, dentro del parámetro `password_md5`. Es el hash que estamos buscando.

```
s: ""
securitytoken: "guest"
do: "addmember"
url: "https://www.linuxquestions.org/"
agree: "1"
password_md5: "81dc9bdb52d04dc20036dbd8313ed055"
passwordconfirm_md5: "81dc9bdb52d04dc20036dbd8313ed055"
day: "0"
month: "0"
year: "0"
username: "CriptoLab"
password: ""
passwordconfirm: ""
email: "criptolab@yopmail.com"
emailconfirm: "criptolab@yopmail.com"
```

Figura 4: Payload del paquete POST.

Para verificar que el hash sea el correcto respecto a MD5, usaremos la página web *md5hashgenerator.com*, dentro de la cual podremos ingresar la contraseña en texto plano y obtener su respectivo hash en MD5, tal como se ve en la figura 5. De esta manera confirmando que el hash corresponde a la contraseña, y que no se ha utilizado ningún salt.

Your String	1234
MD5 Hash	81dc9bdb52d04dc20036dbd8313ed055

Figura 5: Verificación del hash MD5.

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

En la pantalla principal de la página web, en la parte superior derecha, se encuentran el formulario de inicio de sesión, éste tiene la forma vista en la figura 6 más abajo.

Figura 6: Formulario de inicio de sesión.

Dentro del modo inspeccionar elemento, en el documento HTML en la sección del formulario, se encuentra la sección para ingresar la contraseña, la contraseña escrita se ingresa como parámetro para la función `md5hash()`, la cual recibe como parámetros, la contraseña en texto plano y su hash correspondiente, tal como se ve en la figura 7.

```
<form action="https://www.linuxquestions.org/questions/login.php" method="post" onsubmit="md5hash(vb_login_password, vb_login_md5password, vb_login_md5password_utf, 0)"> [event]
<script type="text/javascript" src="/questions/clientscript/vbulletin_md5.js?v=3810b1"></script>
<table cellpadding="0" cellspacing="3" border="0">
  <tbody>
    <tr>
      <td>
        <input type="text" value="User Name" />
      </td>
      <td>
        <input checked="" type="checkbox" /> Remember Me?
      </td>
    </tr>
    <tr>
      <td class="smallfont">Password</td>
      <td>
        <input class="bginput" type="password" style="font-size: 11px" name="vb_login_password" size="10" accesskey="p" tabindex="102">
      </td>
    </tr>
  </tbody>
</table>
```

Figura 7: Inspección de elemento.

De la misma manera que para el registro de usuario, luego de ingresar las credenciales de inicio de sesión, se procede a buscar el paquete POST que guarda las credenciales utilizadas.

Estado	Método	Dominio	Archivo	Iniciador	Tipo	Transferido	Tamaño
200	POST	www.linuxquestions.org	login.php	document	html	3,86 KB	9,32 KB
200	POST	www.google-analytics.com	collect?v=2&tid=G-2K5S76M65W&gtm=45je490v9119862507za200&p=17; js:177 (fetch)			Bloqueado	
200	GET	cdnjs.cloudflare.com	yahoo-dom-event.js?v=3810b1	script	js	cacheado	0 B
200	GET	cdnjs.cloudflare.com	connection-min.js?v=3810b1	script	js	cacheado	0 B
200	GET	www.linuxquestions.org	vbulletin_global.js?v=3810b1	script	js	cacheado	0 B

Figura 8: Inspección de paquete POST generado al momento del login.

Dentro del payload del paquete anteriormente mencionado, se encuentran las credenciales ingresadas, como se había dicho antes, la contraseña en texto plano se encuentra oculta, mientras que solo es visible la contraseña hasheada con MD5. Que resulta ser el mismo hash generado al momento de crear el usuario para la página web.

```
Datos de formulario
vb_login_username: "CriptoLab"
cookieuser: "1"
vb_login_password: ""
s: ""
securitytoken: "guest"
do: "login"
vb_login_md5password: "81dc9bdb52d04dc20036dbd8313ed055"
vb_login_md5password_utf: "81dc9bdb52d04dc20036dbd8313ed055"
```

Figura 9: Payload del paquete POST.

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

Para generar el hash de la contraseña a través de la consola del navegador, se debe buscar la función la cual realiza dicho trabajo. Para ello, dentro del modo inspeccionar elemento, en la sección Debugger o Sources (dependiendo del navegador) realizamos la búsqueda con el filtro “md5hash”, lo que nos llevará a la fuente en donde se encuentra dicha función.

```
=register      vbulletin_md5.js?v=3810b1 X
}function md5hash(B,A,E,C){if(navigator
```

Figura 10: Función de hash MD5.

En la figura anterior no se puede apreciar la función en su totalidad, tan solo su nombre, por lo que se reordenó la función para un mejor entendimiento de ésta, la función ordenada se puede ver en la figura 11.

```
md5hash(B, A, E, C) {
  if (navigator.userAgent.indexOf("Mozilla/") == 0 && parseInt(navigator.appVersion) >= 4) {
    var D = hex_md5(str_to_ent(trim(B.value)));
    A.value = D;
    if (E) {
      D = hex_md5(trim(B.value));
      E.value = D;
    }
    if (!C) {
      B.value = "";
    }
  }
  return true;
}
```

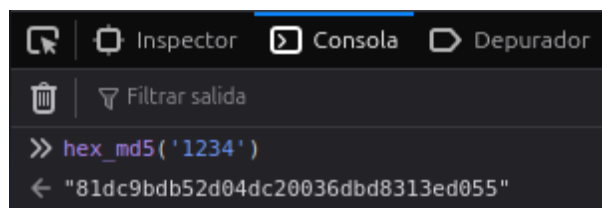
Figura 11: Función de hash MD5 ordenada.

En sí, dentro de la función, se encuentra otra función la cual realiza el hash de la contraseña, ésta se llama `hex_md5` y recibe por parámetro la contraseña ingresada por el usuario. Buscando dicha función se puede encontrar que se compone de la siguiente forma:

```
hex_md5(A){return binl2hex(core_md5(str2binl(A),A.length*chrsz))}
```

Figura 12: Función MD5.

La función vista en la figura 12 será la que utilizaremos en la consola para generar el hash a partir de la contraseña “1234”. Para esto, dentro de la consola se deberá escribir `hex_md5('1234')` y luego apretar enter. El resultado se mostrará inmediatamente y resulta ser el mismo hash mostrado en las figuras 4 y 9. El hash **81dc9bdb52d04dc20036dbd8313ed055** se usará a continuación para el ataque Pass the Hash.



```
>> hex_md5('1234')
<< "81dc9bdb52d04dc20036dbd8313ed055"
```

Figura 13: Generación del hash a través de la consola.

## 2.4. Intercepta el tráfico login con BurpSuite

Lo siguiente es ejecutar el software BurpSuite, con el cual se interceptará tráfico, a través del navegador que provee el programa. Para lograr esto, primero se debe abrir el navegador, acceder en la página web y luego activar la interceptación de tráfico. Después de ingresar las credenciales y clicar en “Log in”, se puede apreciar que BurpSuite ya está capturando tráfico, el paquete que corresponde al que guarda las credenciales es el que tiene por URL <https://www.linuxquestions.org/questions/login.php>, que se puede ver en la figura 14.

Time	Type	Direction	Host	Method	URL
19:28:56 15 Oct 2024	HTTP	→ Request	rum-http-intake.logs.datadoghq.com	POST	https://rum-http-intake.logs.datadoghq.com/v1/input/pubdfe3...
19:28:56 15 Oct 2024	HTTP	→ Request	www.linuxquestions.org	POST	https://www.linuxquestions.org/questions/login.php
19:28:56 15 Oct 2024	HTTP	→ Request	pagead2.googlesyndication.com	POST	https://pagead2.googlesyndication.com/pagead/ping?e=1
19:29:10 15 Oct 2024	HTTP	→ Request	rum-http-intake.logs.datadoghq.com	POST	https://rum-http-intake.logs.datadoghq.com/v1/input/pubdfe3...
19:29:56 15 Oct 2024	HTTP	→ Request	pagead2.googlesyndication.com	POST	https://pagead2.googlesyndication.com/pagead/ping?e=1
19:30:56 15 Oct 2024	HTTP	→ Request	pagead2.googlesyndication.com	POST	https://pagead2.googlesyndication.com/pagead/ping?e=1
19:31:56 15 Oct 2024	HTTP	→ Request	pagead2.googlesyndication.com	POST	https://pagead2.googlesyndication.com/pagead/ping?e=1

Figura 14: Captura de paquetes en BurpSuite.

Dentro del paquete, y al igual que los paquetes vistos en las subsecciones anteriores, se pueden encontrar las credenciales de inicio de sesión.

Name	Value
vb_login_user...	CriptoLab
cookieuser	1
vb_login_pass...	
s	
securitytoken	guest
do	login
vb_login_md5p...	81dc9bdb52d04dc20036dbd8313ed055
vb_login_md5p...	81dc9bdb52d04dc20036dbd8313ed055

Figura 15: Payload del paquete POST con las credenciales.

## 2.5. Realiza el intento de login

Para realizar el ataque de tipo **Pass the Hash**, será necesario utilizar el paquete capturado en el punto anterior, sin embargo, este ataque será realizado con un paquete cuya contraseña ingresada sea incorrecta. Para luego, modificar el campo de hash dentro del paquete, reemplazando el hash de la contraseña incorrecta por el hash de la contraseña correcta, sin ingresar la contraseña correcta directamente. La contraseña incorrecta a utilizar para el ataque será “12345”, cuyo hash MD5 es **827ccb0eea8a706c4c34a16891f84e7b**.

in&vb\_login\_md5password=\$827ccb0eea8a706c4c34a16891f84e7b&vb\_login\_md5password\_utf=\$827ccb0eea8a706c4c34a16891f84e7b&

Figura 16: Campos a modificar durante el ataque PtH.

Al haber dos campos los cuales se deben modificar, no se puede realizar el ataque usando el método Sniper, por lo que se usará el método Cluster bomb al igual que en el laboratorio 2, con la diferencia de que solo habrá una combinación. De esta manera, emulando un ataque con método Sniper, ambos payloads vistos en la figura 17 tendrán el hash MD5 correspondiente a la contraseña “1234”, el cual es **81dc9bdb52d04dc20036dbd8313ed055**.



**Payload sets**

You can define one or more payload sets. The number of payload sets depends on the

Payload set:  Payload count: 1  
 Payload type:  Request count: 1

---

**Payload settings [Simple list]**

This payload type lets you configure a simple list of strings that are used as payloads.

[Pro version only]

Figura 17: Payloads para el ataque.

Después de realizar el ataque, se observan dos requests, el primero corresponde al intento de ingreso con la contraseña incorrecta, y el segundo hace referencia al login con el hash de la contraseña correcta. Por lo tanto, este segundo request es el que se tomará en cuenta.

Request ^	Payload 1	Payload 2	Status code	Response received	Error	Timeout	Length
0			200	273			23642
1	81dc9bdb52d04dc20036dbd8313ed055	81dc9bdb52d04dc20036dbd8313ed055	200	481			10378

Figura 18: Ataque realizado.

La respuesta de la página web al ataque PtH es la esperada por este laboratorio, con un ingreso de sesión exitoso. Dentro del documento HTML se puede apreciar el login exitoso, dando la bienvenida al usuario Criptolab.

```
<blockquote>
  <p>
    &nbsp;
  </p>
  <p>
    <strong>
      Thank you for logging in, CriptoLab.
    </strong>
  </p>
  <p class="smallfont">
    <a href="https://www.linuxquestions.org/questions/index.php">
      Click here if your browser does not automatically redirect you.
    </a>
  </p>
</div>
</blockquote>
```

Figura 19: Response de la página web.

El documento HTML de manera renderizada se puede ver de la siguiente manera:

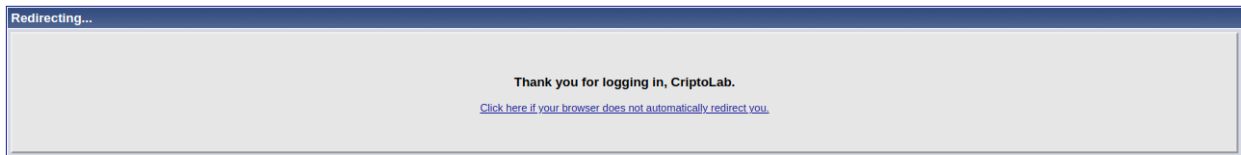


Figura 20: Render de la respuesta de la página web.

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

Es normal que las páginas web tengan políticas de privacidad, donde expongan todas las prácticas de seguridad que realiza la página, ésta no fue la excepción, el link a dicha sección de la página web es el siguiente: <https://www.linuxquestions.org/linux/privacy.html>

Si bien, en esta página las descripciones no son muy explícitas, es posible entender qué hace la web con los datos de los usuarios, lo que se verá a continuación.

Según la política de privacidad de *linuxquestions.org*, este sitio recopila información personal de sus usuarios, incluyendo direcciones de correo electrónico, usuarios, contraseñas e IPs. Sin embargo, no especifica si es que se implementan algoritmos para el hash de contraseñas o demás datos, pero de antemano se sabe que se implementa MD5 para el hash de contraseñas gracias a lo trabajado durante este laboratorio.

Además, se menciona que la web tiene medidas de seguridad contra la pérdida, mal uso o malversación de los datos, pero tampoco menciona cuáles son estas medidas de seguridad. Solo se comparte información que no sea personal con aplicaciones de terceros, información

personal de los usuarios se mantienen exclusivamente para el sitio web.

Si bien se describen las políticas generales de seguridad y privacidad, estas son bastante ambiguas y no detallan los métodos específicos que garantizan la protección de la información personal. Esto podría generar una falta de transparencia respecto a las prácticas de seguridad implementadas, lo que resulta problemático en un entorno donde la seguridad de las contraseñas es crucial.

En particular, sería importante que se mencionara qué algoritmo de hash se utiliza para asegurar las contraseñas, si se aplican técnicas como el salado de hashes, y si las conexiones entre el cliente y el servidor están aseguradas con HTTPS, entre otros detalles técnicos esenciales.

### 2.7. Demuestra 4 conclusiones sobre la seguridad

Después de examinar la implementación de seguridad de linuxquestions.org, se descubrió que hay deficiencias significativas en la gestión de datos confidenciales de los usuarios, particularmente en la protección de contraseñas. El uso del algoritmo MD5 para el hash de contraseñas es uno de los principales problemas, si bien añade una capa de seguridad a la protección de los datos, puede dejar de ser suficiente en algunos aspectos. Este algoritmo puede considerarse obsoleto e inseguro y tiene una serie de vulnerabilidades importantes, incluida su vulnerabilidad a colisiones y ataques de fuerza bruta.

La posibilidad de estar expuesto a ataques de tipo Pass the Hash (PtH) es otra vulnerabilidad importante, tal como se vio durante la práctica de este laboratorio. Los atacantes pueden interceptar o acceder al hash de una contraseña en este tipo de ataque. La falta de claridad sobre la implementación de otras medidas de seguridad complementarias en el lado del cliente o del servidor aumenta este riesgo.

Aunque el sitio utilice el protocolo HTTPS para garantizar la segura transmisión de datos, esto limita la posibilidad de interceptaciones en texto plano. Una estrategia completa de protección de contraseñas, que debe incluir métodos más avanzados de almacenamiento y manejo de credenciales, no puede funcionar con transmisión segura. Además, el hecho de que solo las contraseñas y no otros datos personales sean encriptados demuestra una falta de enfoque completo en la seguridad de la información, lo que abre la posibilidad de brechas.

Por lo tanto, a este sitio web le faltaría agregar mejores prácticas seguras y modernas para la protección de los datos personales de los usuarios que se registran, como cambiar el algoritmo de hash por uno más robusto y agregar salt a estos mismos. Además de que el sitio sea más transparente al momento de exponer las políticas de seguridad empleadas, para que de esta manera el usuario se pueda sentir protegido dentro de las página web.