

# Informe Laboratorio 3

Sección x

Alumno x

e-mail: alumno.contacto@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 . . . . .	4
2.3. Genera el hash de la contraseña desde la consola del navegador . . . . .	5
2.4. Intercepta el tráfico login con BurpSuite . . . . .	6
2.5. Realiza el intento de login . . . . .	6
2.6. Identifica las políticas de privacidad o seguridad . . . . .	8
2.7. Demuestra 4 conclusiones sobre la seguridad . . . . .	8

## 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

En el primer paso de este laboratorio, nos enfocamos en identificar el algoritmo de hash aplicado en el registro de usuarios de un sitio web. Para esto, realizamos un registro en el sitio *free-hack.com*, donde creamos una cuenta utilizando la contraseña LabCripto3. Durante el proceso de registro, mantenemos la consola de desarrollador abierta para observar las solicitudes enviadas al servidor.

Al revisar estas solicitudes, notamos un campo denominado `password_md5`, lo cual sugiere que el sitio podría estar utilizando el algoritmo de hash MD5 para proteger las contraseñas de

los usuarios. Para comprobarlo, encriptamos la contraseña **LabCripto3** utilizando un generador de MD5 en línea, logrando obtener el mismo valor que aparece en el campo `password_md5` de la solicitud.

En la Figura 1, se muestra la solicitud capturada durante el registro, y en la Figura 2, los resultados obtenidos al aplicar el generador de MD5. Esto confirma que el sitio web utiliza el algoritmo MD5 para almacenar contraseñas sin ninguna modificación extra.

```
timezoneoffset: "1"
dst: "2"
options[adminemail]: "1"
options[showemail]: "1"
agree: "1"
s: ""
securitytoken: "guest"
do: "addmember"
url: "https://free-hack.com/"
password_md5: "dd1d59ccf7fdf2e922de43259ca6a2b3"
passwordconfirm_md5: "dd1d59ccf7fdf2e922de43259ca6a2b3"
day: ""
month: ""
```

Figura 1: Captura de la solicitud de registro en *free-hack.com* mostrando el campo `password_md5`.

Use this generator to create an MD5 hash of a string:

LabCripto3

Generate →

Your String	LabCripto3
MD5 Hash	dd1d59ccf7fdf2e922de43259ca6a2b3 <button>Copy</button>

Figura 2: Resultados obtenidos al utilizar un encriptador de MD5 en línea para la contraseña LabCripto3.

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

En esta sección se debe identificar el algoritmo de hash utilizado en los formularios de inicio de sesión. En este caso, al iniciar sesión en la página, no se obtiene información relevante en la solicitud. Sin embargo, al inspeccionar la página, se puede observar cómo en los campos de contraseña aparece explícitamente el uso de MD5. En la figura 3 se puede observar el elemento con las etiquetas que nombran a MD5, identificando así a este algoritmo como el utilizado en este ejemplo.

```

        Benutzername und dein Kennwort in ...ieren <Schaltfläche, um
        ein neues Benutzerkonto anzulegen." accesskey="s">
        whitespace
        <input id="cb_cookieuser_navbar" class="cb_cookieuser_navbar"
        type="checkbox" name="cookieuser" value="1" accesskey="c"
        tabindex="103">
    </div>
</div>
</fieldset>
<input type="hidden" name="s" value="">
<input type="hidden" name="securitytoken"
value="1729841665-9c31791fd2772e02d1e08efbacf2a9d9b9473424">
<input type="hidden" name="do" value="login">
<input type="" name="vb_login_md5password">
<input type="hidden" name="vb_login_md5password_utf">
</form>
▶ <script type="text/javascript"> ... </script>
</li>
</ul>

```

Figura 3: Identificación del uso del algoritmo MD5 en los campos de contraseña.

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

Para generar el hash de la contraseña desde la consola del navegador, es necesario cargar la biblioteca requerida. En este caso, utilizamos la biblioteca **crypto** para generar el hash en MD5. Primero, debemos cargar la biblioteca con un script específico. En la Figura 4 se puede observar el código utilizado para generar el hash. Al compararlo nuevamente con el valor de la Figura 1, podemos determinar que son iguales, lo que indica que la generación de la contraseña hasheada fue exitosa.

```

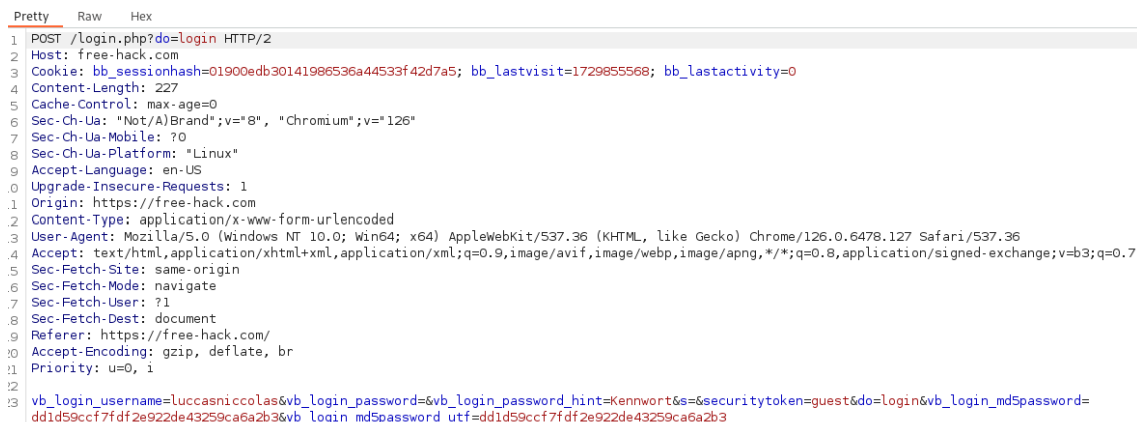
> const script = document.createElement('script');
  script.src = 'https://cdnjs.cloudflare.com/ajax/libs/crypto-js/3.1.9-1/crypto-js.js';
  document.head.appendChild(script);
< ▶ <script src="https://cdnjs.cloudflare.com/ajax/libs/crypto-js/3.1.9-1/crypto-js.js">
> var password = "LabCripto3";
  var hash = CryptoJS.MD5(password).toString();
  console.log(hash);
dd1d59ccf7fdf2e922de43259ca6a2b3

```

Figura 4: Código utilizado para generar el hash MD5 de la contraseña en la consola del navegador.

## 2.4. Intercepta el tráfico login con BurpSuite

Para interceptar el tráfico desde BurpSuite, seguimos el mismo procedimiento que en el laboratorio anterior. Ingresamos las credenciales y, antes de enviarlas, activamos el interceptor. Al enviar las credenciales, obtenemos la siguiente captura:



```

1 POST /login.php?do=login HTTP/2
2 Host: free-hack.com
3 Cookie: bb_sessionhash=01900edb30141986536a44533f42d7a5; bb_lastvisit=1729855568; bb_lastactivity=0
4 Content-Length: 227
5 Cache-Control: max-age=0
6 Sec-Ch-Ua: "Not(A)Brand";v="8", "Chromium";v="126"
7 Sec-Ch-Ua-Mobile: ?0
8 Sec-Ch-Ua-Platform: "Linux"
9 Accept-Language: en-US
10 Upgrade-Insecure-Requests: 1
11 Origin: https://free-hack.com
12 Content-Type: application/x-www-form-urlencoded
13 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.127 Safari/537.36
14 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
15 Sec-Fetch-Site: same-origin
16 Sec-Fetch-Mode: navigate
17 Sec-Fetch-User: ?1
18 Sec-Fetch-Dest: document
19 Referer: https://free-hack.com/
20 Accept-Encoding: gzip, deflate, br
21 Priority: u=0, i
22
23 vb_login_username=luccasniccolas&vb_login_password=6vb_login_password_hint=Kennwort&s=6securitytoken=guest&do=login&vb_login_md5password=
  dd1d59ccf7fdf2e922de43259ca6a2b3&vb_login_md5password_utf=dd1d59ccf7fdf2e922de43259ca6a2b3

```

Figura 5: Tráfico interceptado al ingresar las credenciales

Como se observa en la parte inferior de la captura, se muestra tanto el nombre de usuario como la contraseña en su forma hasheada. Una vez identificados los campos clave, los añadimos al módulo **Intruder**, lo que finaliza este paso del análisis.



```

22 vb_login_username=luccasniccolas&vb_login_password=6vb_login_password_hint=Kennwort&s=6securitytoken=guest&do=login&vb_login_md5password=6dd1d59ccf7fdf2e922de43259ca6a2b3&
23 vb_login_md5password_utf=6dd1d59ccf7fdf2e922de43259ca6a2b3

```

Figura 6: Variables agregadas al módulo Intruder

## 2.5. Realiza el intento de login

Para realizar un intento de login desde BurpSuite, agregamos los payloads con los nuevos hashes que enviaremos. En este caso, solo se cambiará el último carácter del hash. En la Figura 7 se puede observar el nuevo hash que se va a enviar.

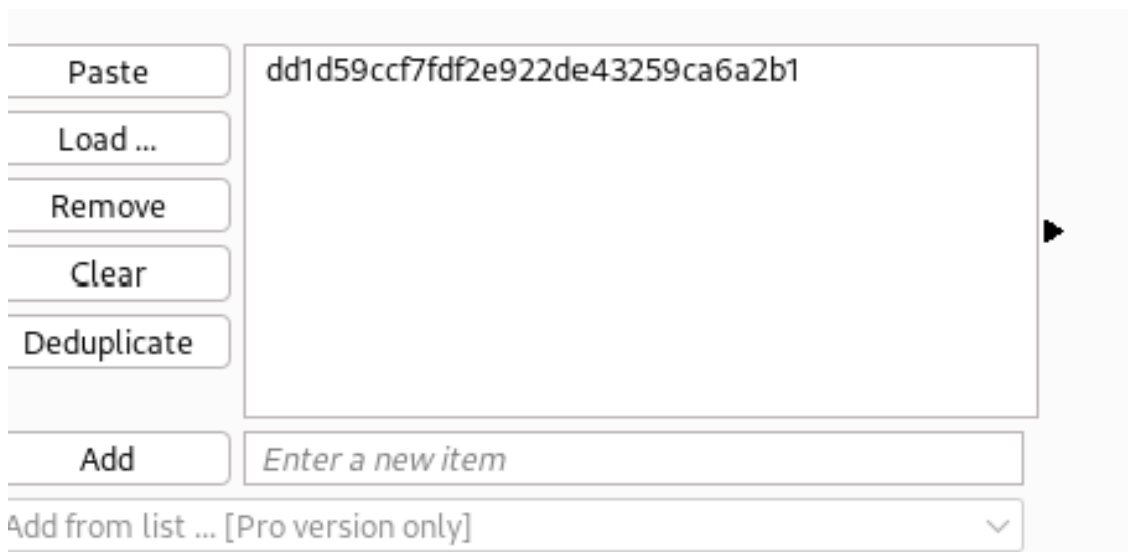


Figura 7: Hash modificado para el intento de login

Luego de enviar la credencial incorrecta, obtenemos la respuesta en BurpSuite. Al ser una página en alemán, es difícil de analizar, pero se pueden observar los formularios de login nuevamente, lo que indica que el intento de inicio de sesión no fue exitoso. En la Figura 8 se muestra la respuesta obtenida en BurpSuite.

```
<!--
<script type="text/javascript" src="clientscript/vbulletin_md5.js?v=425">
</script>
<form id="navbar_loginform" action="login.php?do=login" method="post" onsubmit="md5hash(vb_login_password, vb_login_md5password, vb_login_md5password_utf, 0)">
<fieldset id="logindetails" class="logindetails">
<div>
<div>

<input type="text" class="textbox default-value" name="vb_login_username" id="navbar_username" size="10" accesskey="u" tabindex="101" value="Benutzername" />

<input type="password" class="textbox" tabindex="102" name="vb_login_password" id="navbar_password" size="10" />
<input type="text" class="textbox default-value" tabindex="102" name="vb_login_password_hint" id="navbar_password_hint" size="10" value="Kennwort" style="display:none;" />
<input type="submit" class="loginbutton" tabindex="104" value="Anmelden" title="Gib zur Anmeldung deinen Benutzernamen und dein Kennwort in die dafür vorgesehenen Textfelder ein oder klicke auf die 'Registrieren'-Schaltfläche, um ein neues Benutzerkonto anzulegen." accesskey="s" />
<input type="checkbox" name="cookieuser" value="1" id="cb_cookieuser_navbar" class="cb_cookieuser_navbar" accesskey="c" tabindex="103" />
</div>
</div>
</fieldset>
```

Figura 8: Respuesta del intento de login fallido

A continuación, se muestra la respuesta obtenida al realizar el login con las credenciales correctas.

```
<body>

<div class="standard_error">
  <form class="block vbform" method="post" action="https://free-hack.com/" name="postvarform">
    <h2 class="blockhead">
      Weiterleitung . . .
    </h2>
    <div class="blockbody formcontrols">
      <p class="blockrow restore">
        Danke f  r deine Anmeldung, luccasniccolas.
      </p>
    </div>
    <div class="blockfoot actionbuttons redirect_button">
      <div class="group" id="redirect_button">

        <a href="https://free-hack.com/" class="textcontrol">
          Falls dein Browser dich nicht automatisch weiterleitet, klicke bitte hier.
        </a>

      </div>
    </div>
  </form>
</div>
```

---

Figura 9: Respuesta de un login exitoso

Como se observa, la respuesta es completamente diferente: tiene un formulario, pero no de login o registro, lo que confirma que el intento de inicio de sesi  n fue exitoso con las credenciales correctas. Esto concluye esta secci  n, demostrando la diferencia en las respuestas obtenidas entre un intento de login fallido y uno exitoso.

## 2.6. Identifica las pol  ticas de privacidad o seguridad

Esta p  gina utiliza el algoritmo MD5 para la generaci  n de hashes de contrase  as, un m  todo de seguridad que actualmente es considerado obsoleto debido a su susceptibilidad a ataques de fuerza bruta y colisiones. La elecci  n de MD5 sugiere una pol  tica de seguridad limitada en cuanto a la protecci  n de contrase  as de usuario, ya que no incorpora pr  cticas modernas de seguridad como el uso de algoritmos m  s robustos (por ejemplo, SHA-256 o bcrypt) ni la inclusi  n de un "salt" que dificulte ataques de diccionario.

Adem  s, la p  gina no implementa medidas de seguridad adicionales en la capa de autenticaci  n, como la limitaci  n de intentos fallidos o el uso de CAPTCHAs, lo que puede facilitar ataques de fuerza bruta y scripts autom  ticos para el descubrimiento de contrase  as.

Por otro lado, no se identificaron encabezados HTTP de seguridad adicionales en las respuestas del servidor, como **Content-Security-Policy** o **Strict-Transport-Security**, que ayudan a mitigar ciertos tipos de ataques, tales como inyecci  n de c  digo y ataques de intermediario (*man-in-the-middle*).

Finalmente, la falta de una pol  tica de privacidad visible o accesible en el sitio web sugiere una carencia de transparencia en cuanto al manejo de datos personales y confidenciales de los usuarios. Esto representa un riesgo potencial de exposici  n de la informaci  n de los usuarios y no se alinea con las mejores pr  cticas de protecci  n de datos.

## 2.7. Demuestra 4 conclusiones sobre la seguridad

A partir del an  lisis realizado sobre la seguridad en el sitio web *free-hack.com*, se han identificado los siguientes puntos clave:



1. **Uso del algoritmo MD5 para el hash de contraseñas:** El sitio utiliza el algoritmo MD5 para almacenar contraseñas en formato de hash, como se pudo observar en los campos de contraseña enviados durante el registro e inicio de sesión. Sin embargo, este algoritmo es considerado inseguro debido a su vulnerabilidad a ataques de colisión y fuerza bruta, lo cual expone las contraseñas a posibles compromisos.
2. **Exposición del hash sin modificaciones adicionales:** Durante la inspección de tráfico en BurpSuite, se observó que el hash de la contraseña se transmite sin medidas adicionales de seguridad, como un "salt" (valor aleatorio agregado al hash). Esta ausencia de "salting" facilita aún más los ataques de diccionario y de fuerza bruta, ya que los atacantes pueden predecir y encontrar contraseñas más fácilmente.
3. **Reutilización de hashes en múltiples solicitudes:** En cada intento de inicio de sesión, el sitio utiliza el mismo hash generado desde el cliente sin incluir factores variables (por ejemplo, valores de tiempo o nonce). Esto podría permitir a un atacante reutilizar hashes previamente capturados, facilitando ataques de replay sin necesidad de conocer la contraseña original.
4. **Respuesta del servidor en caso de error visible en las capturas de tráfico:** Cuando el inicio de sesión falla, la respuesta del servidor muestra nuevamente el formulario de login, lo que podría ser utilizado por un atacante para verificar la validez de intentos de login sin ser bloqueado. La ausencia de mecanismos de detección de intentos múltiples, como captchas o bloqueos temporales, sugiere una falta de medidas contra ataques de fuerza bruta.

Estas observaciones indican que el sitio web presenta vulnerabilidades significativas en la gestión de contraseñas y seguridad de autenticación, lo que podría exponer los datos de los usuarios a diversos ataques.