1. **Ataque:** ataque carga de archivo
Vulnerabilidad: Las vulnerabilidades de carga de archivos se producen cuando un servidor web permite a los usuarios cargar archivos en su sistema de archivos sin validar lo suficiente cosas como su nombre, tipo, contenido o tamaño. No hacer cumplir correctamente las restricciones a estos podría significar que incluso una función básica de carga de imágenes se puede utilizar para cargar archivos arbitrarios y potencialmente peligrosos en su lugar. Esto podría incluso incluir archivos de script del lado del servidor que permitan la ejecución remota de código.

Contramedida: OJS guarda todos los archivos subidos por los usuarios en la carpeta ***archivos***, el directorio archivos debe estar ubicado fuera del directorio web, usualmente **public_html**, esto evitará que los archivos potencialmente peligrosos puedan ser ejecutados por el servidor web.

2. Ataque: Xploits para el servidor web (Apache, nginx, IIS, etc.)
Vulnerabilidad: Los atacantes aprovechan vulnerabilidades en el sistema usado como servidor web y pueden llegar a tener control del servidor web mediante algún script php y tener todos los poderes que tiene el usuario con el que se ejecuta el servidor web.
Contramedida: Configuración adecuada de permisos chmod y chown en sistemas *nix evitando que el usuario con el que se ejecuta el servidor web (comúnmente www-data), pueda escribir en los archivos que ejecuta (.php), es decir, que dichos archivos sean de solo lectura para el usuario www-data

3. Ataque: Exposición de datos sensibles
Vulnerabilidad: Cuando no se han realizado las configuraciones necesarias para un ambiente de producción y la aplicación tiene un error o se genera una advertencia y esta se muestra en la propia página, se podría revelar información sensible que podría ser utilizada por los atacantes para vulnerar la aplicación.
Contramedida: En el archivo de configuración de PHP (php.ini) se pueden cambiar las siguientes configuraciones con sus respectivos valores:

display_errors=Off
display_startup_errors=Off
log_errors=On
error_reporting=E_ALL & ~E_DEPRECATED & ~E_STRICT
html_errors=On

La configuración anterior es la recomendada en los comentarios del archivo php.ini

En el archivo de configuración de OJS (config.inc.php) se recomiendan las siguientes configuraciones:

show_stacktrace = Off

display_errors = Off
deprecation_warnings = Off
log_web_service_info = Off

Cualquier configuración diferente a las anteriores deberá ser usada solo en ambientes de desarrollo y con propósitos de depuración para identificar errores.

4. Ataque: Ataque .htaccess
Vulnerabilidad: Este ataque consiste en utilizar una vulnerabilidad del PHP para cambiar los archivos .htaccess y así cambiar el comportamiento del software OJS.
Contramedida: a continuación se presentan un conjunto de medidas para la configuración y protección de este archivo.

- **Evitar el listado de archivos y directorios**

Desde el archivo .htaccess es posible lograr lo anterior agregando estas líneas:

```
Options -Indexes
```

- **Evitar ejecución de archivos PHP**

Una buena práctica es evitar que usuarios no autorizados ejecuten ficheros PHP en aquellas carpetas que son susceptibles a ataques de *malware*. En OJS, por ejemplo, esta acción se realiza en los directorios ***archivos***, directorio donde quedan almacenados los archivos subidos por los usuarios. Para lograr lo anterior se debe crear un archivo .htaccess en la raíz del directorio con la siguiente instrucción:

```
<Files *.php>
deny from all
</Files>
```

- **Proteger archivo config.inc.php**

Este archivo contiene información importante sobre la configuración de OJS, para proteger el archivo config.inc.php se debe agregar un archivo .htaccess con el siguiente código

```
<Files config.inc.php>
order allow, deny
allow from all
```

```
</Files>
```

Proteger archivo .htaccess

Se debe proteger el archivo .htaccess creados, para lo anterior se debe agregar la siguiente instrucción.

```
<Files ~ «^.*\.([Hh][Tt][Aa])»>
order allow, deny
deny from all
satisfy all
</Files>
```

5. Ataque: Ataque de reconocimiento a través del archivo phpinfo
Vulnerabilidad: Es una fase preparatoria donde el hacker utiliza varias técnicas para investigar y recolectar toda la información necesaria de su objetivo antes de lanzar el ataque.
Contramedida: Eliminar o proteger el archivo phpinfo, de manera que un posible atacante no pueda ver información sensible de la configuración del servidor.