1) The remote web server contains a PHP script that is prone to an information disclosure attack.

Many PHP installation tutorials instruct the user to create a PHP file that calls the PHP function 'phpinfo()' for debugging purposes. Various PHP applications may also include such a file. By accessing such a file, a remote attacker can discover a large amount of information about the remote web server, including :

- The username of the user who installed PHP and if they are a SUDO user.
- The IP address of the host.
- The version of the operating system.
- The web server version.
- The root directory of the web server.
- Configuration information about the remote PHP installation.

Disabling phpinfo() won't make your site secure, but will make it slightly more difficult for them.

The 'phpinfo()' should be renamed to something unguessable.

That "phpinfo.php" file has to be removed. Possibly freelancer(website creator) deliberately left it in there as a future hack enabler. Any way is not good, and it should be removed.

2) Web Application Potentially Vulnerable to Clickjacking.

The remote web server does not set an X-Frame-Options response header or a Content-Security-Policy 'frame- ancestors' response header in all content responses. This could potentially expose the site to a clickjacking or UI redress attack, in which an attacker can trick a user into clicking an area of the vulnerable page that is different than what the user perceives the page to be. This can result in a user performing fraudulent or malicious transactions. X-Frame-Options has been proposed by Microsoft as a way to mitigate clickjacking attacks and is currently supported by all major browser vendors.

Content-Security-Policy (CSP) has been proposed by the W3C Web Application Security Working Group, with increasing support among all major browser vendors, as a way to mitigate clickjacking and other attacks. The 'frame- ancestors' policy directive restricts which sources can embed the protected resource.

Note that while the X-Frame-Options and Content-Security-Policy response headers are not the only mitigations for clickjacking, they are currently the most reliable methods that can be detected through automation. Therefore, this plugin may produce false positives if other mitigation strategies (e.g., frame-busting JavaScript) are deployed or if the page does not perform any security-sensitive transactions.

Follow the content security policy guideline to secure the website from cross scripting and clickjacking.

Protection against clickjacking can be added to browser desktop and mobile versions by installing the NoScript add-on.

Web site owners can protect their users against UI redressing (frame based clickjacking) on the server side by including a framekiller JavaScript snippet in those pages they do not want to be included inside frames from different sources.

`X-Frame-Options` which offered a partial protection against clickjacking and prevents the page's content from being rendered by another site when using the frame or iframe HTML tags.