ZAP Scanning Report

Виконали група 3#45

Sites: http://oliva.in.ua https://oliva.in.ua

Generated on ср, 18 січ. 2023 16:34:25

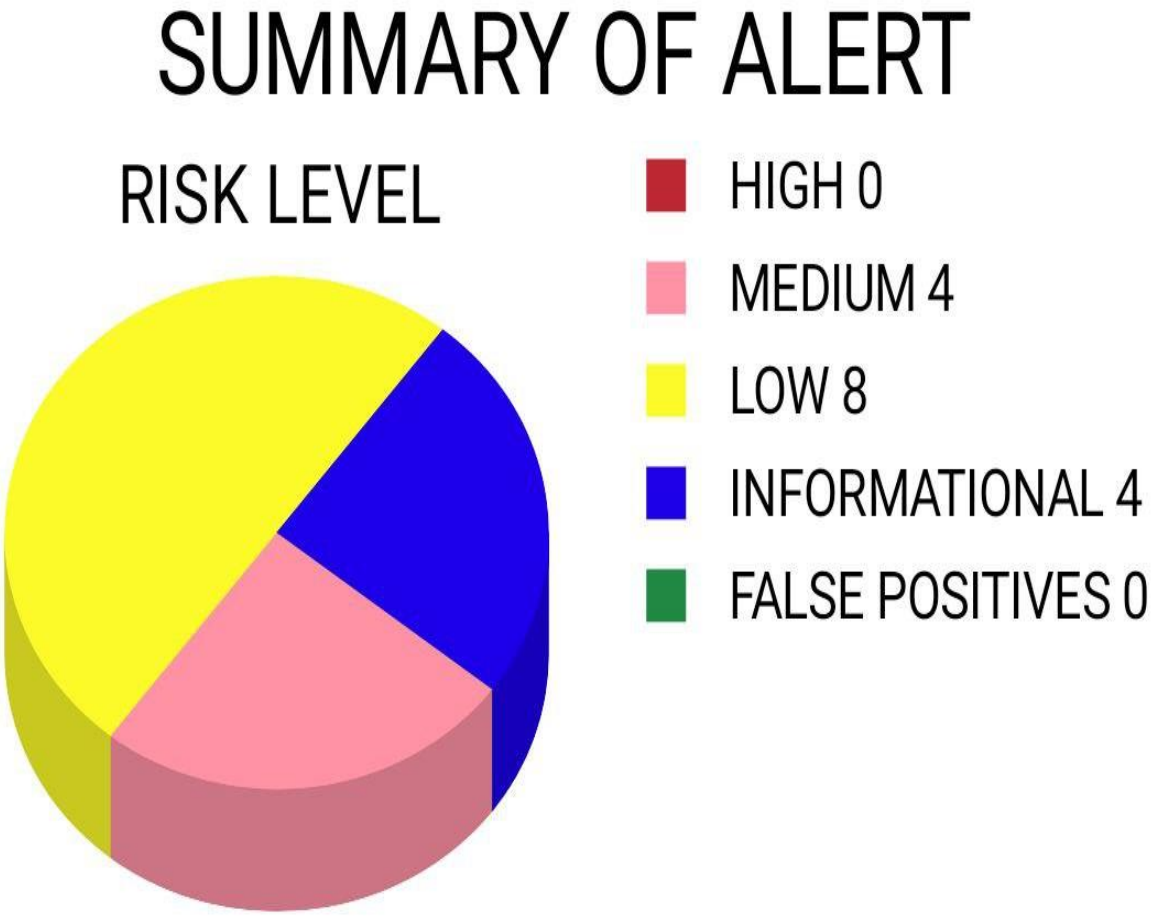
Summary of Alerts

Risk Level	Number of Alerts
High	0
Medium	4
Low	8
Informational	4
False Positives:	0

Оповіщення

Name	Risk Level	Number of Instances
Absence of Anti-CSRF Tokens	Medium	5749
Content Security Policy (CSP) Header Not Set	Medium	1453
Missing Anti-clickjacking Header	Medium	1406
Vulnerable JS Library	Medium	1
Application Error Disclosure	Low	2
Cookie No HttpOnly Flag	Low	1689
Cookie Without Secure Flag	Low	4098
Cookie without SameSite Attribute	Low	4102
Cross-Domain JavaScript Source File Inclusion	Low	7210
Strict-Transport-Security Header Not Set	Low	2834
Timestamp Disclosure - Unix	Low	6786
X-Content-Type-Options Header Missing	Low	2463
Information Disclosure - Suspicious Comments	Informational	1495
Modern Web Application	Informational	1572
Re-examine Cache-control Directives	Informational	1592
User Controllable HTML Element Attribute (Potential XSS)	Informational	4219

The details of the vulnerability of the site https://oliva.in.ua/ are presented in the diagram



The information is shortened to Description of the problem and its Solution.

Alert Detail

Medium	Absence of Anti-CSRF Tokens
Description	No Anti-CSRF tokens were found in a HTML submission form. A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL/form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) exploits the trust that a user has for a web site. Like XSS, CSRF attacks are not necessarily cross-site, but they can be. Cross-site request forgery is also known as CSRF, XSRF, one-click attack, session riding, confused deputy, and sea surf. CSRF attacks are effective in a number of situations, including: * The victim has an active session on the target site. * The victim is authenticated via HTTP auth on the target site. * The victim is on the same local network as the target site. CSRF has primarily been used to perform an action against a target site using the victim's privileges, but recent techniques have been discovered to disclose information by gaining access to the response. The risk of information disclosure is dramatically increased when the target site is

	vulnerable to XSS, because XSS can be used as a platform for CSRF, allowing the attack to operate within the bounds of the same-origin policy.
Solution	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid. For example, use anti-CSRF packages such as the OWASP CSRFGuard. Phase: Implementation Ensure that your application is free of cross-site scripting issues, because most CSRF defenses can be bypassed using attacker-controlled script. Phase: Architecture and Design Generate a unique nonce for each form, place the nonce into the form, and verify the nonce upon receipt of the form. Be sure that the nonce is not predictable (CWE-330). Note that this can be bypassed using XSS. Identify especially dangerous operations. When the user performs a dangerous operation, send a separate confirmation request to ensure that the user intended to perform that operation. Note that this can be bypassed using XSS. Use the ESAPI Session Management control. This control includes a component for CSRF. Do not use the GET method for any request that triggers a state change.

	Phase: Implementation Check the HTTP Referer header to see if the request originated from an expected page. This could break legitimate functionality, because users or proxies may have disabled sending the Referer for privacy reasons.

Medium	Content Security Policy (CSP) Header Not Set
Description	Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.
Solution	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header, to achieve optimal browser support: "Content-Security-Policy" for Chrome 25+, Firefox 23+ and Safari 7+, "X-Content-Security-Policy" for Firefox 4.0+ and Internet Explorer 10+, and "X-WebKit-CSP" for Chrome 14+ and Safari 6+.

Medium	Missing Anti-clickjacking Header
Description	The response does not include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame-Options to protect against 'ClickJacking' attacks.
Solution	Modern Web browsers support the Content-Security-Policy and X-Frame-Options HTTP headers. Ensure one of them is set on all web pages returned by your site/app. If you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to

	use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. Alternatively consider implementing Content Security Policy's "frame-ancestors" directive.

Medium	**Vulnerable JS Library**
Description	The identified library jquery, version 2.2.0 is vulnerable.
Solution	Please upgrade to the latest version of jquery.

Low	**Application Error Disclosure**
Description	This page contains an error/warning message that may disclose sensitive information like the location of the file that produced the unhandled exception. This information can be used to launch further attacks against the web application. The alert could be a false positive if the error message is found inside a documentation page.
Solution	Review the source code of this page. Implement custom error pages. Consider implementing a mechanism to provide a unique error reference/identifier to the client (browser) while logging the details on the server side and not exposing them to the user.

Low	**Cookie No HttpOnly Flag**
Description	A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site. If this is a session cookie then session hijacking may be possible.
Solution	Ensure that the HttpOnly flag is set for all cookies.

Low	**Cookie Without Secure Flag**
Description	A cookie has been set without the secure flag, which means

	that the cookie can be accessed via unencrypted connections.
Solution	Whenever a cookie contains sensitive information or is a session token, then it should always be passed using an encrypted channel. Ensure that the secure flag is set for cookies containing such sensitive information.

Low	Cookie without SameSite Attribute
Description	A cookie has been set without the SameSite attribute, which means that the cookie can be sent as a result of a 'cross-site' request. The SameSite attribute is an effective counter measure to cross-site request forgery, cross-site script inclusion, and timing attacks.
Solution	Ensure that the SameSite attribute is set to either 'lax' or ideally 'strict' for all cookies.

Low	Cross-Domain JavaScript Source File Inclusion
Description	The page includes one or more script files from a third-party domain.
Solution	Ensure JavaScript source files are loaded from only trusted sources, and the sources can't be controlled by end users of the application.

Low	Strict-Transport-Security Header Not Set
Description	HTTP Strict Transport Security (HSTS) is a web security policy mechanism whereby a web server declares that complying user agents (such as a web browser) are to interact with it using only secure HTTPS connections (i.e. HTTP layered over TLS/SSL). HSTS is an IETF standards track protocol and is specified in RFC 6797.
Solution	Ensure that your web server, application server, load balancer, etc. is configured to enforce Strict-Transport-Security.

Low	Timestamp Disclosure - Unix
Description	A timestamp was disclosed by the application/web server - Unix
Solution	Manually confirm that the timestamp data is not sensitive, and that the data cannot be aggregated to disclose exploitable patterns.

Low	X-Content-Type-Options Header Missing
Description	The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.
Solution	Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages. If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.

Informational	Information Disclosure - Suspicious Comments
Description	The response appears to contain suspicious comments which may help an attacker. Note: Matches made within script blocks or files are against the entire content not only comments.
Solution	Remove all comments that return information that may help an attacker and fix any underlying problems they refer to.

Informational	Modern Web Application
Description	The application appears to be a modern web application. If

	you need to explore it automatically then the Ajax Spider may well be more effective than the standard one.
Solution	This is an informational alert and so no changes are required.

Informational	Re-examine Cache-control Directives
Description	The cache-control header has not been set properly or is missing, allowing the browser and proxies to cache content. For static assets like css, js, or image files this might be intended, however, the resources should be reviewed to ensure that no sensitive content will be cached.
Solution	For secure content, ensure the cache-control HTTP header is set with "no-cache, no-store, must-revalidate". If an asset should be cached consider setting the directives "public, max-age, immutable".

Informational	User Controllable HTML Element Attribute (Potential XSS)
Description	This check looks at user-supplied input in query string parameters and POST data to identify where certain HTML attribute values might be controlled. This provides hot-spot detection for XSS (cross-site scripting) that will require further review by a security analyst to determine exploitability.
Solution	Validate all input and sanitize output it before writing to any HTML attributes.