



TutorLingua CASA DAST Scan Report

Google Cloud Application Security Assessment - Dynamic Application Security Testing report for tutorlingua.co. This report is generated for submission to Google CASA.

Sites: <https://firefox-settings-attachments.cdn.mozilla.net>
<https://tutorlingua.co>

Generated on Wed, 17 Dec 2025 19:29:56

ZAP Version: 2.17.0

ZAP by [Checkmarx](#)

Summary of Alerts

Risk Level	Number of Alerts
High	0
Medium	2
Low	2
False Positives:	0

Insights

Level	Reason	Site	Description	Statistic
Medium	Exceeded High	https://tutorlingua.co	Percentage of authentication failures	100 %
Low	Warning		ZAP errors logged - see the zap.log file for details	2
Low	Warning		ZAP warnings logged - see the zap.log file for details	1
Info	Informational	https://firefox-settings-attachments.cdn.mozilla.net	Percentage of endpoints with content type text/plain	100 %
Info	Informational	https://firefox-settings-attachments.cdn.mozilla.net	Percentage of endpoints with method GET	100 %
Info	Informational	https://firefox-settings-attachments.cdn.mozilla.net	Count of total endpoints	1
			Percentage of responses	

Info	Informational	https://tutorlingua.co	with status code 2xx	1 %
Info	Informational	https://tutorlingua.co	Percentage of responses with status code 3xx	99 %
Info	Informational	https://tutorlingua.co	Percentage of endpoints with content type text/html	20 %
Info	Informational	https://tutorlingua.co	Percentage of endpoints with content type text/plain	80 %
Info	Informational	https://tutorlingua.co	Percentage of endpoints with method GET	80 %
Info	Informational	https://tutorlingua.co	Percentage of endpoints with method POST	20 %
Info	Informational	https://tutorlingua.co	Count of total endpoints	5
Info	Exceeded Low	https://tutorlingua.co	Percentage of slow responses	7 %

Alerts

Name	Risk Level	Number of Instances
Content Security Policy (CSP) Header Not Set	Medium	1
Missing Anti-clickjacking Header	Medium	1
Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)	Low	1
X-Content-Type-Options Header Missing	Low	1

Alert Detail

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.
URL	https://tutorlingua.co
Node Name	https://tutorlingua.co
Method	GET
Attack	

Evidence	
Other Info	
Instances	1
Solution	Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.
Reference	https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/CSP https://cheatsheetseries.owasp.org/cheatsheets/Content_Security_Policy_Cheat_Sheet.html https://www.w3.org/TR/CSP/ https://w3c.github.io/webappsec-csp/ https://web.dev/articles/csp https://caniuse.com/#feat=contentsecuritypolicy https://content-security-policy.com/
CWE Id	693
WASC Id	15
Plugin Id	10038

Medium	Missing Anti-clickjacking Header
Description	The response does not protect against 'ClickJacking' attacks. It should include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame-Options.
URL	https://tutorlingua.co
Node Name	https://tutorlingua.co
Method	GET
Attack	
Evidence	
Other Info	
Instances	1
Solution	<p>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.</p> <p>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.</p>
Reference	https://developer.mozilla.org/en-US/docs/Web/HTTP/Reference/Headers/X-Frame-Options
CWE Id	1021
WASC Id	15
Plugin Id	10020

Low	Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)
Description	The web/application server is leaking information via one or more "X-Powered-By" HTTP response headers. Access to such information may facilitate attackers identifying other frameworks/components your web application is reliant upon and the vulnerabilities such components may be subject to.
URL	https://tutorlingua.co
Node Name	https://tutorlingua.co
Method	GET
Attack	

Evidence	X-Powered-By: Next.js
Other Info	
Instances	1
Solution	Ensure that your web server, application server, load balancer, etc. is configured to suppress "X-Powered-By" headers.
Reference	https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application_Security_Testing/01-Information_Gathering/08-Fingerprint_Web_Application_Framework https://www.troyhunt.com/shhh-dont-let-your-response-headers/
CWE Id	497
WASC Id	13
Plugin Id	10037

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.
URL	https://tutorlingua.co
Node Name	https://tutorlingua.co
Method	GET
Attack	
Evidence	
Other Info	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.
Instances	1
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.
Reference	https://learn.microsoft.com/en-us/previous-versions/windows/internet-explorer/ie-developer/compatibility/gg622941(v=vs.85) https://owasp.org/www-community/Security-Headers
CWE Id	693
WASC Id	15
Plugin Id	10021