Vulnerability Scan Report

ZAP Scanning Report

Sites: https://www.google.com https://google.com

Generated on Tue, 15 Apr 2025 15:25:23

Summary of Alerts

Risk Level	Number of Alerts
High	0
Medium	1
Low	4
Informational	6
False Positives:	0

Summary of Sequences

For each step: result (Pass/Fail) - risk (of highest alert(s) for the step, if any).

Alerts

Name	Risk Level	Number of Instances
Content Security Policy (CSP) Header Not Set	Medium	1
Cookie Without Secure Flag	Low	1
Cookie without SameSite Attribute	Low	1
Strict-Transport-Security Header Not Set	Low	5
X-Content-Type-Options Header Missing	Low	1
Charset Mismatch (Header Versus Meta Content-Type Charset)	Informational	1

Content Security Policy (CSP) Report-Only Header Found	Informational	1
Loosely Scoped Cookie	Informational	1
Re-examine Cache-control Directives	Informational	1
Retrieved from Cache	Informational	1
Session Management Response Identified	Informational	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://www.google.com/
	Method	GET
	Parameter	
	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/Security/CSP/Introducing_Content_Security_Policy 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
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.
URL	https://www.google.com/
Method	GET
Parameter	NID
Attack	
Evidence	Set-Cookie: NID
Other Info	
Instances	1
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.
Reference	https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/06-Se
CWE Id	614
WASC Id	13
Plugin Id	10011
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.
URL	https://www.google.com/
Method	GET
Parameter	NID
Attack	
Evidence	Set-Cookie: NID
Other Info	
Instances	1
Solution	Ensure that the SameSite attribute is set to either 'lax' or ideally 'strict' for all cookies.
Reference	https://tools.ietf.org/html/draft-ietf-httpbis-cookie-same-site
CWE Id	1275

WASC Id	13
Plugin Id	10054
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.
URL	https://google.com
Method	GET
Parameter	
Attack	
Evidence	
Other Info	
URL	https://google.com/
Method	GET
Parameter	
Attack	
Evidence	
Other Info	
URL	https://google.com/robots.txt
Method	GET
Parameter	
Attack	
Evidence	
Other Info	
URL	https://google.com/sitemap.xml
Method	GET
Parameter	
Attack	

Evidence	
Other Info	
URL	https://www.google.com/
Method	GET
Parameter	
Attack	
Evidence	
Other Info	
Instances	5
Solution	Ensure that your web server, application server, load balancer, etc. is configured to enforce Strict-Transport-Security.
Reference	https://cheatsheetseries.owasp.org/cheatsheets/HTTP_Strict_Transport_Security_Cheat_Sheet.html https://owasp.org/www-community/Security_Headers https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security https://caniuse.com/stricttransportsecurity https://datatracker.ietf.org/doc/html/rfc6797
CWE Id	319
WASC Id	15
Plugin Id	10035
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://www.google.com/
Method	GET
Parameter	x-content-type-options
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/g https://owasp.org/www-community/Security_Headers
CWE Id		693
WASC Id		15
Plugin Id		10021
Informational		**Charset Mismatch (Header Versus Meta Content-Type Charset)**
Description		This check identifies responses where the HTTP Content-Type header declares a charset different from the charset defined by the body of the HTML or XML. When there's a charset mismatch between the HTTP header and content body Web browsers can be forced into an undesirable content-sniffing mode to determine the content's correct character set. An attacker could manipulate content on the page to be interpreted in an encoding of their choice. For example, if an attacker can control content at the beginning of the page, they could inject script using UTF-7 encoded text and manipulate some browsers into interpreting that text.
URL		https://www.google.com/
	Method	GET
	Parameter	
	Attack	
	Evidence	
	Other Info	There was a charset mismatch between the HTTP Header and the META content-type encoding declarations: [ISO-8859-1] and [UTF-8] do not match.
Instances		1
Solution		Force UTF-8 for all text content in both the HTTP header and meta tags in HTML or encoding declarations in XML.
Reference		https://code.google.com/p/browsersec/wiki/Part2#Character_set_handling_and_detection
CWE Id		436

WASC Id	15
Plugin Id	90011
Informational	**Content Security Policy (CSP) Report-Only Header Found**
Description	The response contained a Content-Security-Policy-Report-Only header, this may indicate a work-in-progress implementation, or an oversight in promoting pre-Prod to Prod, etc. 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://www.google.com/
Method	GET
Parameter	
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://www.w3.org/TR/CSP2/ https://w3c.github.io/webappsec-csp/ https://caniuse.com/#feat=contentsecuritypolicy https://content-security-policy.com/
CWE Id	693
WASC Id	15
Plugin Id	10038
Informational	**Loosely Scoped Cookie**
Description	Cookies can be scoped by domain or path. This check is only concerned with domain scope.The domain scope applied to a cookie determines which domains can access it. For example, a cookie can be scoped strictly to a subdomain e.g. www.nottrusted.com, or loosely scoped to a parent domain e.g. nottrusted.com. In the latter case, any subdomain of nottrusted.com can access the cookie. Loosely scoped cookies are common in mega-applications like google.com and live.com. Cookies set from a subdomain like app.foo.bar are transmitted only to that domain by the browser. However, cookies scoped to a parent-level domain may be transmitted to the parent, or any subdomain of the parent.

URL		https://www.google.com/
	Method	GET
	Parameter	
	Attack	
	Evidence	

	Other Info	The origin domain used for comparison was: www.google.com
		AEC=AVcja2ciDnLsJIPqlJTZnbr6zqJTsdEQkwgRyp8Ej13TSDcNI0729bnhCNg
		NID=523=PBGgFv4opNkO1lB1O-uXE1zLrbXhw2jxt5rskd7yPasSKR4U_PbnvIDLidXjwY0OwU-qfspuFDS

Instances		1
Solution		Always scope cookies to a FQDN (Fully Qualified Domain Name).
Reference		https://tools.ietf.org/html/rfc6265#section-4.1 https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/06-Se https://code.google.com/p/browsersec/wiki/Part2#Same-origin_policy_for_cookies
CWE Id		565
WASC Id		15
Plugin Id		90033
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.
URL		https://www.google.com/
	Method	GET
	Parameter	cache-control
	Attack	
	Evidence	private, max-age=0
	Other Info	
Instances		1
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".

Reference	https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html#web-content-c https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Cache-Control https://grayduck.mn/2021/09/13/cache-control-recommendations/
CWE Id	525
WASC Id	13
Plugin Id	10015
Informational	**Retrieved from Cache**
Description	The content was retrieved from a shared cache. If the response data is sensitive, personal or user-specific, this may result in sensitive information being leaked. In some cases, this may even result in a user gaining complete control of the session of another user, depending on the configuration of the caching components in use in their environment. This is primarily an issue where caching servers such as "proxy" caches are configured on the local network. This configuration is typically found in corporate or educational environments, for instance.
URL	https://google.com/robots.txt
Method	GET
Parameter	
Attack	
Evidence	Age: 1228
Other Info	The presence of the 'Age' header indicates that a HTTP/1.1 compliant caching server is in use.
Instances	1
Solution	Validate that the response does not contain sensitive, personal or user-specific information. If it does, consider the use of the following HTTP response headers, to limit, or prevent the content being stored and retrieved from the cache by another user: Cache-Control: no-cache, no-store, must-revalidate, private Pragma: no-cache Expires: 0 This configuration directs both HTTP 1.0 and HTTP 1.1 compliant caching servers to not store the response, and to not retrieve the response (without validation) from the cache, in response to a similar request.
Reference	https://tools.ietf.org/html/rfc7234 https://tools.ietf.org/html/rfc7231 https://www.rfc-editor.org/rfc/rfc9110.html
CWE Id	

WASC Id		
Plugin Id		10050
Informational		**Session Management Response Identified**
Description		The given response has been identified as containing a session management token. The 'Other Info' field contains a set of header tokens that can be used in the Header Based Session Management Method. If the request is in a context which has a Session Management Method set to "Auto-Detect" then this rule will change the session management to use the tokens identified.
URL		https://www.google.com/
	Method	GET
	Parameter	NID
	Attack	
	Evidence	523=PBGgFv4opNkO1lB1O-uXE1zLrbXhw2jxt5rskd7yPasSKR4U_PbnvIDLidXjwY0OwU-qfspuFDS6zJH
	Other Info	cookie:NID cookie:AEC
Instances		1
Solution		This is an informational alert rather than a vulnerability and so there is nothing to fix.
Reference		https://www.zaproxy.org/docs/desktop/addons/authentication-helper/session-mgmt-id
CWE Id		
WASC Id		
Plugin Id		10112

Sequence Details

With the associated active scan results.

Report generated by VirtuesTech Security Scanner