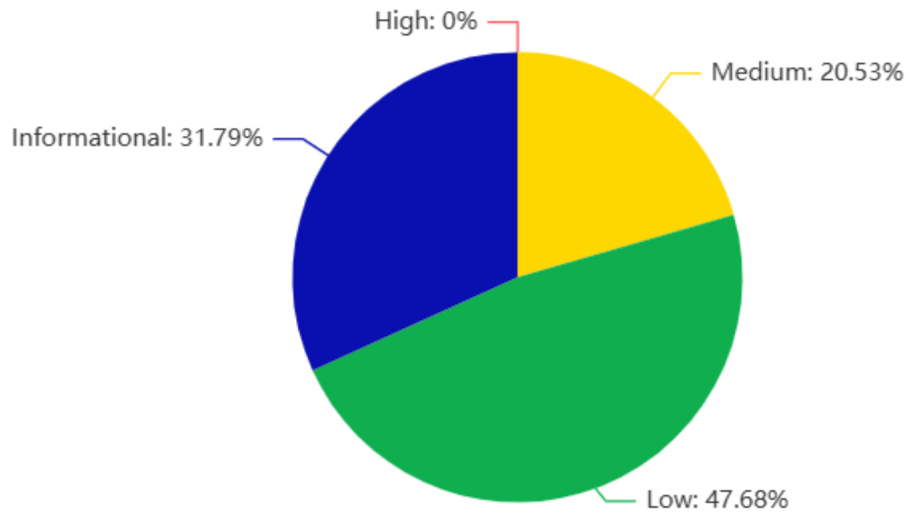
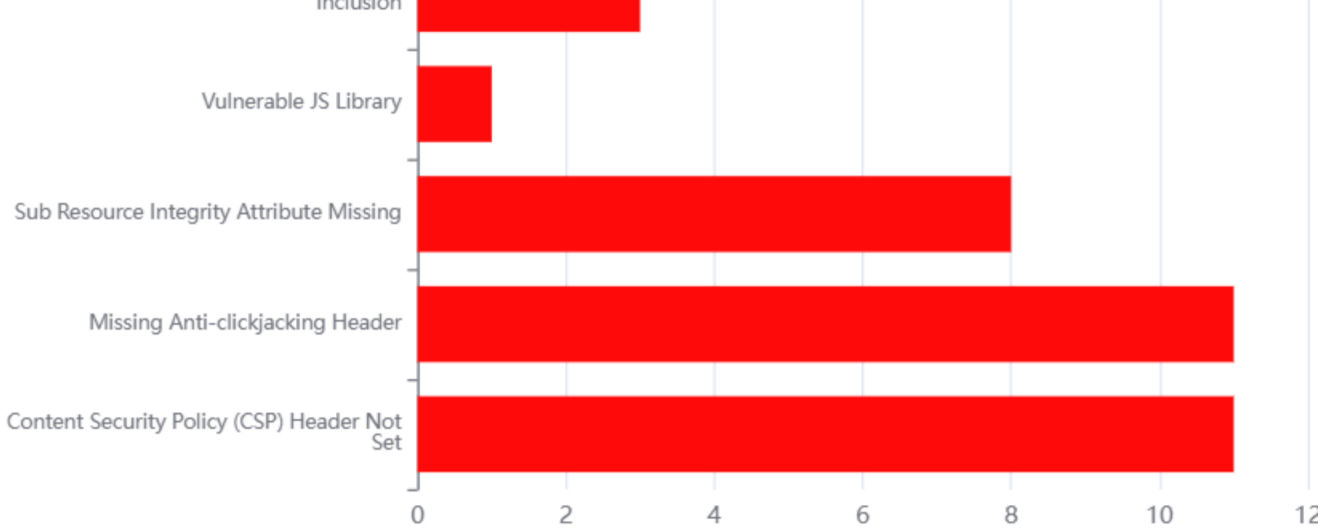
Security Test Report

Vulnerability Severity



Vulnerability Count



AI Summary 1/1:

Report Metadata

- **Generated On:** Thu, 3 Apr 2025 21:41:09
- **Scanned Site:** https://dml.ece.ubc.ca
- **Total Vulnerabilities Identified:** 20

Top 3 Critical Vulnerabilities

Content Security Policy (CSP) Header Not Set

- **Risk Level:** Medium (High)
- **Count:** 11
- **Exploitability:** High
- **Impact:** CSP helps detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks.
- **Why is it critical?** It's a major concern because it leaves the site vulnerable to XSS and data injection attacks.
- **Mitigation Strategy:** Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.
- **References:** https://developer.mozilla.org/en-US/docs/Web/Security/CSP/Introducing_Content_Security_Policy

Missing Anti-clickjacking Header

- **Risk Level:** Medium (Medium)
- **Count:** 11
- **Exploitability:** Medium
- **Impact:** The response does not protect against 'ClickJacking' attacks.
- **Why is it critical?** It's a major concern because it leaves the site vulnerable to ClickJacking attacks.
- **Mitigation Strategy:** Ensure one of them is set on all web pages returned by your site/app.
- **References:** https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options

Sub Resource Integrity Attribute Missing

- **Risk Level:** Medium (High)
- **Count:** 8

- **Exploitability:** High
- **Impact:** The integrity attribute prevents an attacker who have gained access to this server from injecting a malicious content.
- **Why is it critical?** It's a major concern because it leaves the site vulnerable to malicious content injection.
- **Mitigation Strategy:** Provide a valid integrity attribute to the tag.
- **References:** https://developer.mozilla.org/en-US/docs/Web/Security/Subresource_Integrity

Remaining Vulnerabilities

Vulnerable JS Library

Cross-Domain JavaScript Source File Inclusion

Information Disclosure - Debug Error Messages

Insufficient Site Isolation Against Spectre Vulnerability

Permissions Policy Header Not Set

Secure Pages Include Mixed Content

Server Leaks Version Information via "Server" HTTP Response Header Field

Strict-Transport-Security Header Not Set

X-Content-Type-Options Header Missing

Content-Type Header Missing

Information Disclosure - Suspicious Comments

Modern Web Application

Re-examine Cache-control Directives

Storable and Cacheable Content

Additional Security Recommendations

- Implement a Web Application Firewall (WAF) to detect and prevent common web attacks.
- Use a Content Security Policy (CSP) to define which sources of content are allowed to be executed within a web page.
- Set the X-Frame-Options header to prevent ClickJacking attacks.
- Use the Strict-Transport-Security header to enforce HTTPS connections.
- Set the X-Content-Type-Options header to prevent MIME-sniffing attacks.
- Use the Cache-Control header to control caching behavior.

Final Risk Assessment

- **Overall Security Rating:** Medium
- **Key Areas of Concern:** Content Security Policy (CSP) Header Not Set, Missing Anti-clickjacking Header, Sub Resource Integrity Attribute Missing
- **Critical Fixes Required:** Implement a Web Application Firewall (WAF), set the X-Frame-Options header, use the Strict-Transport-Security header
- **Suggested Next Steps:** Review and implement the recommended security measures, perform regular

security audits and vulnerability assessments.

Report

DAST Test Configurations
program name: ZAP
version: 2.16.0
generated: Thu, 3 Apr 2025 21:41:09

Site General Information
name: https://dml.ece.ubc.ca
host: dml.ece.ubc.ca
port: 443
ssl: true

Name	Impact
Content Security Policy (CSP) Header Not Set Risk: Medium (High)	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. **Possible Solution:** Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header. **References:** 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/
Missing Anti-clickjacking Header Risk: Medium (Medium)	The response does not protect against 'ClickJacking' attacks. It should include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame-Options. **Possible 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. **References:** https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
Sub Resource Integrity Attribute Missing Risk: Medium (High)	The integrity attribute is missing on a script or link tag served by an external server. The integrity tag prevents an attacker who have gained access to this server from injecting a malicious content. **Possible Solution:** Provide a valid integrity attribute to the tag. **References:** https://developer.mozilla.org/en-US/docs/Web/Security/Subresource_Integrity
Vulnerable JS Library Risk: Medium (Medium)	The identified library appears to be vulnerable. **Possible Solution:** Upgrade to the latest version of the affected library. **References:** https://owasp.org/Top10/A06_2021-Vulnerable_and_Outdated_Components/
Cross-Domain	

Cross-Domain JavaScript Source File Inclusion	The page includes one or more script files from a third-party domain.
Information Disclosure - Debug Error Messages	The response appeared to contain common error messages returned by platforms such as ASP.NET, and Web-servers such as IIS and Apache. You can configure the list of common debug messages.
Insufficient Site Isolation Against Spectre Vulnerability Risk: Low (Medium)	Cross-Origin-Resource-Policy header is an opt-in header designed to counter side-channels attacks like Spectre. Resource should be specifically set as shareable amongst different origins. **Possible Solution:** Ensure that the application/web server sets the Cross-Origin-Resource-Policy header appropriately, and that it sets the Cross-Origin-Resource-Policy header to 'same-origin' for all web pages. 'same-site' is considered as less secured and should be avoided. If resources must be shared, set the header to 'cross-origin'. If possible, ensure that the end user uses a standards-compliant and modern web browser that supports the Cross-Origin-Resource-Policy header (https://caniuse.com/mdn-http_headers_cross-origin-resource-policy). **References:** https://developer.mozilla.org/en-US/docs/Web/HTTP/Cross-Origin_Resource_Policy
Permissions Policy Header Not Set Risk: Low (Medium)	Permissions Policy Header is an added layer of security that helps to restrict from unauthorized access or usage of browser/client features by web resources. This policy ensures the user privacy by limiting or specifying the features of the browsers can be used by the web resources. Permissions Policy provides a set of standard HTTP headers that allow website owners to limit which features of browsers can be used by the page such as camera, microphone, location, full screen etc. **Possible Solution:** Ensure that your web server, application server, load balancer, etc. is configured to set the Permissions-Policy header. **References:** https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Permissions-Policy

	https://developer.chrome.com/blog/feature-policy/ https://scotthelme.co.uk/a-new-security-header-feature-policy/ https://w3c.github.io/webappsec-feature-policy/ https://www.smashingmagazine.com/2018/12/feature-policy/
Secure Pages Include Mixed Content Risk: Low (Medium)	The page includes mixed content, that is content accessed via HTTP instead of HTTPS. **Possible Solution:** A page that is available over SSL/TLS must be comprised completely of content which is transmitted over SSL/TLS. The page must not contain any content that is transmitted over unencrypted HTTP. This includes content from third party sites. **References:** https://cheatsheetseries.owasp.org/cheatsheets/Transport_Layer_Protection_Cheat_Sheet.html
Server Leaks Version Information via "Server" HTTP Response Header Field Risk: Low (High)	The web/application server is leaking version information via the "Server" HTTP response header. Access to such information may facilitate attackers identifying other vulnerabilities your web/application server is subject to. **Possible Solution:** Ensure that your web server, application server, load balancer, etc. is configured to suppress the "Server" header or provide generic details. **References:** https://httpd.apache.org/docs/current/mod/core.html#servertokens https://learn.microsoft.com/en-us/previous-versions/msp-n-p/ff648552(v=pandp.10) https://www.troyhunt.com/shhh-dont-let-your-response-headers/
	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

Strict-Transport-Security Header Not Set Risk: Low (High)	**Possible Solution:** Ensure that your web server, application server, load balancer, etc. is configured to enforce Strict-Transport-Security. **References:** 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
X-Content-Type-Options Header Missing Risk: Low (Medium)	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. **Possible 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. **References:** 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
Content-Type Header Missing Risk:	The Content-Type header was either missing or empty. **Possible Solution:** Ensure each page is setting the specific and appropriate content-type value for the content being delivered.

Risk: Informational (Medium)	**References:** https://learn.microsoft.com/en-us/previous-versions/windows/internet-explorer/ie-developer/compatibility/gg622941(v=vs.85)
Information Disclosure - Suspicious Comments	The response appears to contain suspicious comments which may help an attacker.
Modern Web Application	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.
Re-examine Cache-control Directives Risk: Informational (Low)	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. **Possible 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". **References:** https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html#web-content-caching https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Cache-Control https://grayduck.mn/2021/09/13/cache-control-recommendations/
	The response contents are storable by caching components such as proxy servers, and may be retrieved directly from the cache, rather than from the origin server by the caching servers, in response to similar requests from other users. 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 "shared" caching servers such as "proxy" caches are configured on the local network. This configuration is typically found in corporate or educational environments, for instance

Storable and Cacheable Content Risk: Informational (Medium)	found in corporate or educational environments, for instance. **Possible 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. **References:** https://datatracker.ietf.org/doc/html/rfc7234 https://datatracker.ietf.org/doc/html/rfc7231 https://www.w3.org/Protocols/rfc2616/rfc2616-sec13.html