# ZAP Scanning Report

## Site: http://localhost:3000

### Generated on Mon, 2 Oct 2023 23:46:28

### ZAP Version: 2.13.0

### Summary of Alerts

| Risk Level | Number of Alerts |
|---|---|
| High | 0 |
| Medium | 4 |
| Low | 3 |
| Informational | 2 |

### Alerts

| Name | Risk Level | Number of Instances |
|---|---|---|
| CSP: Wildcard Directive | Medium | 2 |
| Content Security Policy (CSP) Header Not Set | Medium | 2 |
| Cross-Domain Misconfiguration | Medium | 9 |
| Missing Anti-clickjacking Header | Medium | 2 |
| Cross-Domain JavaScript Source File Inclusion | Low | 2 |
| Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s) | Low | 9 |
| X-Content-Type-Options Header Missing | Low | 6 |
| Information Disclosure - Suspicious Comments | Informational | 4 |
| Modern Web Application | Informational | 2 |

### Alert Detail

| Medium | CSP: Wildcard Directive |
|---|---|
| Description | Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) 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 | http://localhost:3000/sitemap.xml |
| Method | GET |
| Attack | |
| Evidence | default-src 'none' |
| Other | The following directives either allow wildcard sources (or ancestors), are not defined, or are overly broadly defined: frame-ancestors, form-action The directive(s): frame-ancestors, |

| | |
|---|---|
| Info | form-action are among the directives that do not fallback to default-src, missing/excluding them is the same as allowing anything. |
| URL | http://localhost:3000/src/Component/Images/Coloration.png |
| Method | GET |
| Attack | |
| Evidence | default-src 'none' |
| Other Info | The following directives either allow wildcard sources (or ancestors), are not defined, or are overly broadly defined: frame-ancestors, form-action The directive(s): frame-ancestors, form-action are among the directives that do not fallback to default-src, missing/excluding them is the same as allowing anything. |
| Instances | 2 |
| Solution | Ensure that your web server, application server, load balancer, etc. is properly configured to set the Content-Security-Policy header. |
| Reference | http://www.w3.org/TR/CSP2/<br>http://www.w3.org/TR/CSP/<br>http://caniuse.com/#search=content+security+policy<br>http://content-security-policy.com/<br>https://github.com/shapesecurity/salvation<br>https://developers.google.com/web/fundamentals/security/csp#policy_applies_to_a_wide_variety_of_resources |
| CWE Id | 693 |
| WASC Id | 15 |
| Plugin Id | 10055 |

| **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 | http://localhost:3000 |
| Method | GET |
| Attack | |
| Evidence | |
| Other Info | |
| URL | http://localhost:3000/ |
| Method | GET |
| Attack | |
| Evidence | |
| Other Info | |
| Instances | 2 |
| 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<br>https://cheatsheetseries.owasp.org/cheatsheets/Content_Security_Policy_Cheat_Sheet.html<br><br>http://www.w3.org/TR/CSP/ |

| | |
|---|---|
| | http://w3c.github.io/webappsec/specs/content-security-policy/csp-specification.dev.html<br>http://www.html5rocks.com/en/tutorials/security/content-security-policy/<br>http://caniuse.com/#feat=contentsecuritypolicy<br>http://content-security-policy.com/ |
| CWE Id | 693 |
| WASC Id | 15 |
| Plugin Id | 10038 |

| Medium | Cross-Domain Misconfiguration |
|---|---|
| Description | Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server |
| URL | http://localhost:3000 |
| Method | GET |
| Attack | |
| Evidence | Access-Control-Allow-Origin: * |
| Other Info | The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing. |
| URL | http://localhost:3000/ |
| Method | GET |
| Attack | |
| Evidence | Access-Control-Allow-Origin: * |
| Other Info | The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing. |
| URL | http://localhost:3000/favicon.png |
| Method | GET |
| Attack | |
| Evidence | Access-Control-Allow-Origin: * |
| Other Info | The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing. |
| URL | http://localhost:3000/logo192.png |
| Method | GET |
| Attack | |
| Evidence | Access-Control-Allow-Origin: * |
| Other Info | The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing. |
| URL | http://localhost:3000/manifest.json |

| | |
|---|---|
| Method | GET |
| Attack | |
| Evidence | Access-Control-Allow-Origin: * |
| Other Info | The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing. |
| URL | http://localhost:3000/robots.txt |
| Method | GET |
| Attack | |
| Evidence | Access-Control-Allow-Origin: * |
| Other Info | The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing. |
| URL | http://localhost:3000/sitemap.xml |
| Method | GET |
| Attack | |
| Evidence | Access-Control-Allow-Origin: * |
| Other Info | The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing. |
| URL | http://localhost:3000/src/Component/Images/Coloration.png |
| Method | GET |
| Attack | |
| Evidence | Access-Control-Allow-Origin: * |
| Other Info | The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing. |
| URL | http://localhost:3000/static/js/bundle.js |
| Method | GET |
| Attack | |
| Evidence | Access-Control-Allow-Origin: * |
| Other Info | The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing. |
| Instances | 9 |
| | Ensure that sensitive data is not available in an unauthenticated manner (using IP address white-listing, for instance). |

| | |
|---|---|
| Solution | Configure the "Access-Control-Allow-Origin" HTTP header to a more restrictive set of domains, or remove all CORS headers entirely, to allow the web browser to enforce the Same Origin Policy (SOP) in a more restrictive manner. |
| Reference | https://vulncat.fortify.com/en/detail?id=desc.config.dotnet.html5_overly_permissive_cors_policy |
| CWE Id | 264 |
| WASC Id | 14 |
| Plugin Id | 10098 |

| **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. |
| URL | http://localhost:3000 |
| Method | GET |
| Attack | |
| Evidence | |
| Other Info | |
| URL | http://localhost:3000/ |
| Method | GET |
| Attack | |
| Evidence | |
| Other Info | |
| Instances | 2 |
| 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.<br><br>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. |
| Reference | https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options |
| CWE Id | 1021 |
| WASC Id | 15 |
| Plugin Id | 10020 |

| **Low** | **Cross-Domain JavaScript Source File Inclusion** |
|---|---|
| Description | The page includes one or more script files from a third-party domain. |
| URL | http://localhost:3000 |
| Method | GET |
| Attack | |
| Evidence | <script src="https://accounts.google.com/gsi/client"></script> |
| Other Info | |
| URL | http://localhost:3000/ |
| Method | GET |
| | |

| | |
|---|---|
| Attack | |
| Evidence | <script src="https://accounts.google.com/gsi/client"></script> |
| Other Info | |
| Instances | 2 |
| Solution | Ensure JavaScript source files are loaded from only trusted sources, and the sources can't be controlled by end users of the application. |
| Reference | |
| CWE Id | 829 |
| WASC Id | 15 |
| Plugin Id | 10017 |

| **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 | http://localhost:3000 |
| Method | GET |
| Attack | |
| Evidence | X-Powered-By: Express |
| Other Info | |
| URL | http://localhost:3000/ |
| Method | GET |
| Attack | |
| Evidence | X-Powered-By: Express |
| Other Info | |
| URL | http://localhost:3000/favicon.png |
| Method | GET |
| Attack | |
| Evidence | X-Powered-By: Express |
| Other Info | |
| URL | http://localhost:3000/logo192.png |
| Method | GET |
| Attack | |
| Evidence | X-Powered-By: Express |
| Other Info | |
| URL | http://localhost:3000/manifest.json |
| Method | GET |
| Attack | |
| Evidence | X-Powered-By: Express |
| Other | |

| | |
|---|---|
| Info | |
| URL | http://localhost:3000/robots.txt |
| Method | GET |
| Attack | |
| Evidence | X-Powered-By: Express |
| Other Info | |
| URL | http://localhost:3000/sitemap.xml |
| Method | GET |
| Attack | |
| Evidence | X-Powered-By: Express |
| Other Info | |
| URL | http://localhost:3000/src/Component/Images/Coloration.png |
| Method | GET |
| Attack | |
| Evidence | X-Powered-By: Express |
| Other Info | |
| URL | http://localhost:3000/static/js/bundle.js |
| Method | GET |
| Attack | |
| Evidence | X-Powered-By: Express |
| Other Info | |
| Instances | 9 |
| Solution | Ensure that your web server, application server, load balancer, etc. is configured to suppress "X-Powered-By" headers. |
| Reference | http://blogs.msdn.com/b/varunm/archive/2013/04/23/remove-unwanted-http-response-headers.aspx<br>http://www.troyhunt.com/2012/02/shhh-dont-let-your-response-headers.html |
| CWE Id | 200 |
| 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 | http://localhost:3000 |
| Method | GET |
| Attack | |
| Evidence | |
| | This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still |

| | |
|---|---|
| Other Info | 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. |
| URL | http://localhost:3000/ |
| 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. |
| URL | http://localhost:3000/favicon.png |
| 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. |
| URL | http://localhost:3000/logo192.png |
| 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. |
| URL | http://localhost:3000/manifest.json |
| 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. |
| URL | http://localhost:3000/robots.txt |
| 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 | 6 |
| 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.<br><br>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 | http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx<br>https://owasp.org/www-community/Security_Headers |
| CWE Id | 693 |
| WASC Id | 15 |
| Plugin Id | 10021 |

| **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. |
| URL | http://localhost:3000 |
| Method | GET |
| Attack | |
| Evidence | from |
| Other Info | The following pattern was used: \bFROM\b and was detected in the element starting with: "<!-- Notice the use of in the tags above. It will be replaced with the URL of the `public` folder during the buil", see evidence field for the suspicious comment/snippet. |
| URL | http://localhost:3000 |
| Method | GET |
| Attack | |
| Evidence | user |
| Other Info | The following pattern was used: \bUSER\b and was detected in the element starting with: "<!-- manifest.json provides metadata used when your web app is installed on a user's mobile device or desktop. See", see evidence field for the suspicious comment/snippet. |
| URL | http://localhost:3000/ |
| Method | GET |
| Attack | |
| Evidence | from |
| Other Info | The following pattern was used: \bFROM\b and was detected in the element starting with: "<!-- Notice the use of in the tags above. It will be replaced with the URL of the `public` folder during the buil", see evidence field for the suspicious comment/snippet. |
| URL | http://localhost:3000/ |
| Method | GET |
| Attack | |
| Evidence | user |
| Other Info | The following pattern was used: \bUSER\b and was detected in the element starting with: "<!-- manifest.json provides metadata used when your web app is installed on a user's mobile device or desktop. See", see evidence field for the suspicious comment/snippet. |
| Instances | 4 |
| Solution | Remove all comments that return information that may help an attacker and fix any underlying problems they refer to. |
| Reference | |
| CWE Id | 200 |
| WASC Id | 13 |
| Plugin Id | 10027 |

| **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. |

| | |
|---|---|
| URL | http://localhost:3000 |
| Method | GET |
| Attack | |
| Evidence | <script src="https://accounts.google.com/gsi/client"></script> |
| Other Info | No links have been found while there are scripts, which is an indication that this is a modern web application. |
| URL | http://localhost:3000/ |
| Method | GET |
| Attack | |
| Evidence | <script src="https://accounts.google.com/gsi/client"></script> |
| Other Info | No links have been found while there are scripts, which is an indication that this is a modern web application. |
| Instances | 2 |
| Solution | This is an informational alert and so no changes are required. |
| Reference | |
| CWE Id | |
| WASC Id | |
| Plugin Id | 10109 |