Content Security Policy

https://campus.barracuda.com/doc/95259569/

Content Security Policy (CSP) is a security layer added to detect cross-site scripting (XSS), clickjacking, and other code injection attacks. CSP allows the user to take strict control over all resources that load into the web browser while accessing your application.

Configuring a Content Security Policy involves adding the content security policy HTTP header to a web page and providing values to control the type of resources the user agent is allowed to load for that page.

Add Content Security Policy

1. Go to **WEBSITES > Client Side Protection**.
2. On the **Client Side Protection** section and from the **Actions** column, click the drop-down list and the select **Add CSP** for the rule. The **Content Security Policy** window appears.
3. Specify values for the following fields:
    - **CSP Policy Name** - Enter a name for the CSP policy.
    - **Status** - Select to set the status of the CSP rule. Set to **On** to display the status of the CSP rule.
    - **Mode** - Select a mode of the CSP.
        - **Block** - Use this mode to block the CSP rule.
        - **Report Only** - Use this mode to report only the CSP rule.
    - **CSP Policy Action** - Select any one action that you want to perform for the CSP policy.
        - **Create Policy** - Provides a wizard with all CSP directives to help the administrator configure the appropriate CSP policy
        - **Upload Policy** - Use this capability to upload the CSP policy if you already have it defined.
    - **CSP Directives -** CSP Directives are a consistent set of directives that tell the browser what content sources should be trusted and which ones should be blocked. Using the defined policies, you can restrict the browser content to eliminate many common injection vectors and significantly reduce the risk of XSS attacks.
    - **Default Source** - Select the source you want to set as default.
        - **Space separated list of hosts** - Provide host names separated by space. For example, `http://*.`example.com mail.example.com:443 https://store.example.com separated by space.
        - **Script Source** - Specifies valid sources for JavaScript. This includes not only URLs loaded directly into script elements, but also inline script event handlers (onclick) and XSLT style sheets that can trigger script execution.
            - **Script Source Elements** - Specifies valid sources for JavaScript script elements.

        - **Script Source for Script Requests and Script Blocks** - Specifies valid sources for JavaScript script requests and blocks.
        - **Script Source Attribute** - Specifies valid sources for JavaScript inline event handlers.
- **Style Source** - Specifies valid sources for style sheets.
    - **Style Source** - Specifies valid sources for style sheets.
    - **Style Source, Except for Styles Defined in Inline Attributes** – Specifies valid sources for style sheets, except for styles defined in attributes.
    - **Style Source in Attributes** – Specifies valid sources for style sheets in attributes.
- **Other Sources**
    - **Image Source** - Specifies valid sources for images and favicons.
    - **Font Source** - Specifies valid sources for fonts loaded using @font-face.
    - **Connect Source** - Specifies valid sources for fetch, XMLHttpRequest, WebSocket, and EventSource connections.
    - **Media Source** - Specifies valid sources for the audio and video elements.
    - **Object Source** - Specifies valid sources for the object embed and applet elements.
    - **Prefetch Source** - Restricts the URLs from which resources may be prefetched or prerendered.
    - **Child Source** - Specifies valid sources for elements such as frame and iframe.
    - **Worker Source** - Specifies valid sources for Worker, SharedWorker or ServiceWorker.
    - **Manifest Source** - Specifies which manifest can be applied to the resource. This directive falls back to default-src if not specified.
    - **Plugin Types** – Provide the plugin types the browser may invoke.
- **Content/Plugin/Sandbox**
    - **Sand Box** - Applies restrictions to a page including the prevention of pop-ups, plugins, scripts and enforcing a same-origin policy.
    - **Base URI** - Specifies URIs that a user agent may use as the document's base URL for relative URIs.
    - **Block All Mixed Content** - Forces a user agent to load all assets over HTTPS, even if the URL specifies HTTP when the page is loaded using HTTPS. It is ignored in a Report-Only policy.
- **Form/Frames/Request**
    - **Form Action** - Specifies locations that can be used for form submissions.
    - **Frames Ancestors** - Specifies parents that may embed a page using elements such as frame and iframe. It replaces the X-Frame-Options header
    - **Frame Source** - Specifies valid sources for elements such as frame and iframe
    - **Upgrade Insecure Requests** - Forces a user agent to load all assets over HTTPS, even if the URL specifies HTTP, when the page is loaded using HTTPS. It is ignored in a Report-Only policy**.**
- **Reporting**
    - **Report URI** - Specifies the URI that the user agent will use to POST a JSON-

formatted violation report to the CSP to be violated.
- **Report To** - Specifies a token that the user agent will use to look up the reporting group in the report-to header.

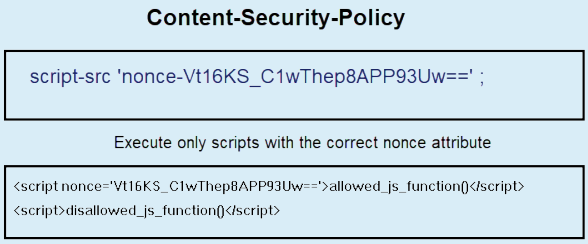
▪ **Deprecated**
- **Disown Opener** - Ensures that a resource disowns its opener when navigated to.

4. Click **Save** to save the configuration.

When “Include Nonce” is selected in the CSP policy, the ‘Content-Security-Policy’ / ‘Content-Security-Policy-Report-Only’ header specifies the ‘nonce-<RANDOM>'' that needs to be included by all JavaScripts present in the web page in order to allow the execution. For security reasons, the value of ‘nonce’ changes for every request and is non-predictable.

In general, Content Security Policy advocates for removing all inline JavaScripts from the webpage. However, if removing all such instances in one go is not possible, it provides support for ‘nonce’.

Example:



The ‘script’ tag has included an additional attribute named ‘nonce’ whose value is the same as that of the value specified in the policy. In this way, the browser is assured that the ‘inline’ content has indeed come from the server of this web page and is not appearing because of any injection attack.

Edit/Delete CSP

For the CSP whose values you want to edit, perform the following steps:

1. Select the CSP you want to edit.
2. From the **Actions** column, choose **Edit/Delete** from the drop-down list to update or delete CSP.
3. Make changes to the CSP values and then click **Save**.

Figures