Documentation of Removing script-src data Directive from Content Security

Introduction

The purpose of this document is to record the process and methods used to remove the `data:` directive from the Content Security Policy (CSP) of our organization. The `data:` directive allows data URLs to be used in content, which can pose security risks. This documentation will cover various approaches attempted to achieve this goal, along with their outcomes.

Overview of CSP and `data:` Directive

Content Security Policy (CSP) is a security feature that helps prevent various attacks, including Cross-Site Scripting (XSS) and data injection attacks. The `data:` directive in CSP allows the use of data URLs. For enhanced security, removing this directive can reduce the attack surface.

Default CSP

- Current Default CSP Our application works fine as expected with `script-src data: *`

```
img-src data: blob: *;
default-src blob: *;
script-src data: * 'nonce-dummy' 'unsafe-eval';
style-src 'unsafe-inline' *;
font-src data: *"
```

Updating CSP Headers without` `script-src data:`

- The first approach involves directly modifying the CSP headers to remove the `data:` directive and observing the immediate impact.

Steps

1. Access the server configuration where CSP headers are defined.
2. Remove the `data:` directive from the CSP header.

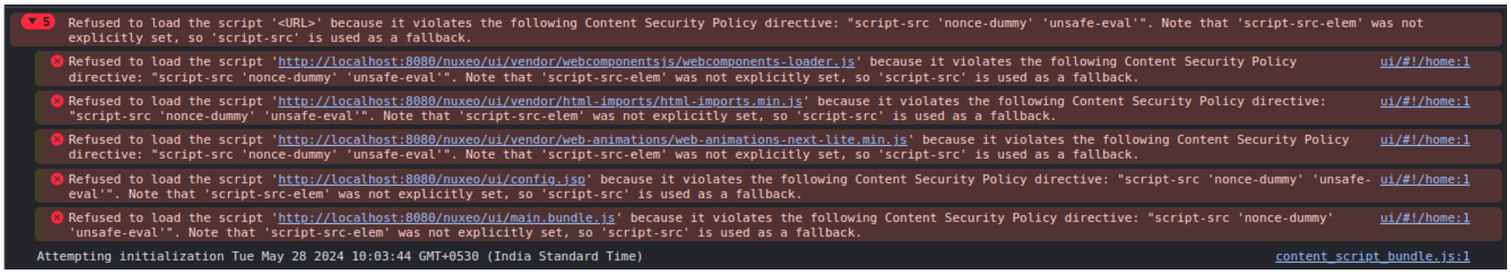
```
img-src data: blob: *;
default-src blob: *;
script-src self 'nonce-dummy' 'unsafe-eval';
style-src 'unsafe-inline' *;
font-src data: *"
```

4. Deploy the changes in a testing environment.
5. Monitor the blocked resources and functionality issues.

Outcome

- Application breaks and show blank page.
- Shows below errors in `html-imports.min.js`.

```
Refused to load the script 'http://<domain>/nuxeo/ui/vendor/webcomponentsjs/webcomponents-loader.js' because it violates the following Content Security Policy directive: "script-src 'nonce-dummy' 'unsafe-eval'". Note that 'script-src-elem' was not explicitly set, so 'script-src' is used as a fallback.
```



```
▼5 Refused to load the script '<URL>' because it violates the following Content Security Policy directive: "script-src 'nonce-dummy' 'unsafe-eval'". Note that 'script-src-elem' was not
   explicitly set, so 'script-src' is used as a fallback.
  ⊗ Refused to load the script 'http://localhost:8080/nuxeo/ui/vendor/webcomponentsjs/webcomponents-loader.js' because it violates the following Content Security Policy      ui/#!/home:1
    directive: "script-src 'nonce-dummy' 'unsafe-eval'". Note that 'script-src-elem' was not explicitly set, so 'script-src' is used as a fallback.
  ⊗ Refused to load the script 'http://localhost:8080/nuxeo/ui/vendor/html-imports/html-imports.min.js' because it violates the following Content Security Policy directive:   ui/#!/home:1
    "script-src 'nonce-dummy' 'unsafe-eval'". Note that 'script-src-elem' was not explicitly set, so 'script-src' is used as a fallback.
  ⊗ Refused to load the script 'http://localhost:8080/nuxeo/ui/vendor/web-animations/web-animations-next-lite.min.js' because it violates the following Content Security Policy   ui/#!/home:1
    directive: "script-src 'nonce-dummy' 'unsafe-eval'". Note that 'script-src-elem' was not explicitly set, so 'script-src' is used as a fallback.
  ⊗ Refused to load the script 'http://localhost:8080/nuxeo/ui/config.jsp' because it violates the following Content Security Policy directive: "script-src 'nonce-dummy' 'unsafe- ui/#!/home:1
    eval'". Note that 'script-src-elem' was not explicitly set, so 'script-src' is used as a fallback.
  ⊗ Refused to load the script 'http://localhost:8080/nuxeo/ui/main.bundle.js' because it violates the following Content Security Policy directive: "script-src 'nonce-dummy'   ui/#!/home:1
    'unsafe-eval'". Note that 'script-src-elem' was not explicitly set, so 'script-src' is used as a fallback.
Attempting initialization Tue May 28 2024 10:03:44 GMT+0530 (India Standard Time)                                                             content_script_bundle.js:1
```

here are the list of files, shown in the above picture.

1. `nuxeo/ui/vendor/webcomponentsjs/webcomponents-loader.js`
2. `nuxeo/ui/vendor/html-imports/html-imports.min.js`
3. `nuxeo/ui/vendor/web-animations/web-animations-next-lite.min.js`
4. `nuxeo/ui/config.jsp`
5. `nuxeo/ui/main.bundle.js`

Note:

- nuxeo-web-ui >= 2021 version is using Polymer 3
- Polymer 3 versions no longer include the HTML imports polyfill, and have been developed to work with ES6 modules.
- But still in webpack polyfill we are using `html-imports.min.js`

Approach 1: nonce-based

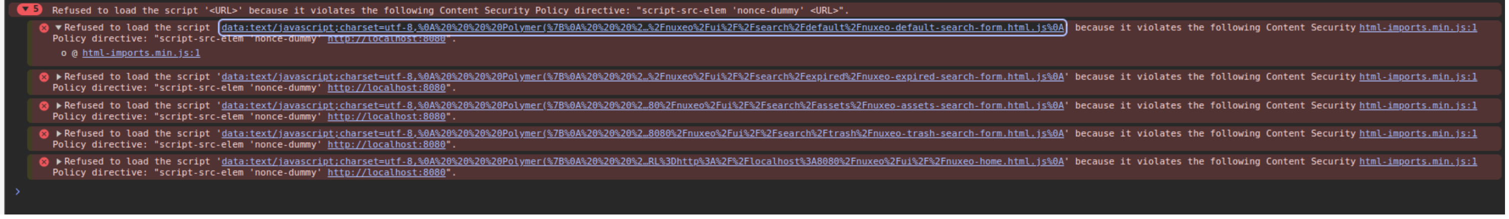
In this approach, we enhance the security of our web UI by implementing nonce-based Content Security Policy (CSP). We achieve this by adding a `nonce` attribute to all script tags within our web UI including above 5 files.

PR: https://github.com/nuxeo/nuxeo-web-ui/pull/2245 Ref: https://content-security-policy.com/nonce/

example:

```
<script nonce="dummy">....</script>
```

Outcome:



```
▼5 Refused to load the script '<URL>' because it violates the following Content Security Policy directive: "script-src-elem 'nonce-dummy' <URL>".
  ▼Refused to load the script data:text/javascript;charset=utf-8,%0A%20%20%20%20Polymer(%7B%0A%20%20%20%2…%2Fnuxeo%2Fui%2F%2Fsearch%2Fdefault%2Fnuxeo-default-search-form.html.js%0A because it violates the following Content Security html-imports.min.js:1
   Policy directive: "script-src-elem 'nonce-dummy' http://localhost:8080".
     o @ html-imports.min.js:1
  ▶Refused to load the script 'data:text/javascript;charset=utf-8,%0A%20%20%20%20Polymer(%7B%0A%20%20%20%2…%2Fnuxeo%2Fui%2F%2Fsearch%2Fexpired%2Fnuxeo-expired-search-form.html.js%0A' because it violates the following Content Security html-imports.min.js:1
   Policy directive: "script-src-elem 'nonce-dummy' http://localhost:8080".
  ▶Refused to load the script 'data:text/javascript;charset=utf-8,%0A%20%20%20%20Polymer(%7B%0A%20%20%20%2…80%2Fnuxeo%2Fui%2F%2Fsearch%2Fassets%2Fnuxeo-assets-search-form.html.js%0A' because it violates the following Content Security html-imports.min.js:1
   Policy directive: "script-src-elem 'nonce-dummy' http://localhost:8080".
  ▶Refused to load the script 'data:text/javascript;charset=utf-8,%0A%20%20%20%20Polymer(%7B%0A%20%20%20%2…8080%2Fnuxeo%2Fui%2F%2Fsearch%2Ftrash%2Fnuxeo-trash-search-form.html.js%0A' because it violates the following Content Security html-imports.min.js:1
   Policy directive: "script-src-elem 'nonce-dummy' http://localhost:8080".
  ▶Refused to load the script 'data:text/javascript;charset=utf-8,%0A%20%20%20%20Polymer(%7B%0A%20%20%20%2…RL%3Dhttp%3A%2F%2Flocalhost%3A8080%2Fnuxeo%2Fui%2F%2Fnuxeo-home.html.js%0A' because it violates the following Content Security html-imports.min.js:1
   Policy directive: "script-src-elem 'nonce-dummy' http://localhost:8080".
>
```

here are the list of new files, shown in the above picture.

1. nuxeo-home
2. nuxeo-default-search-form
3. nuxeo-browser
4. nuxeo-trash-search-form
5. nuxeo-assets-search-form
6. nuxeo-expired-search-form

Approach 2: 'strict-dynamic'

Using the `'strict-dynamic'` keyword in the Content Security Policy (CSP) is a powerful approach to mitigate the risk of XSS (Cross-Site Scripting) attacks while allowing for dynamic script execution from trusted sources.

Trusted sources

- Ensure that all dynamically generated script tags (e.g., created via `document.createElement('script')` or `eval()`) are injected only from trusted sources.
- These trusted sources should be explicitly whitelisted in the CSP header or be included in the `'self'` directive if they originate from the same origin.

```
img-src data: blob: *;
default-src blob: *;
script-src 'nonce-dummy' 'unsafe-eval' 'strict-dynamic';
style-src 'unsafe-inline' *;
font-src data: *
```

Outcome:

- Same as approach 1

Final Analysis:

By Combining both Approach 1 & 2 we can remove data directive from our default csp and also `unsafe-eval`

```
img-src data: blob: *;
default-src blob: *;
script-src 'nonce-dummy' 'strict-dynamic';
style-src 'unsafe-inline' *;
font-src data: *
```