parallax / jsPDF Public

Code | Issues 139 | Pull requests 37 | Actions | Projects 1 | Security

New issue

Jump to bottom

XSS vulnerability in html method #2795

Closed HackbrettXXX opened this issue on Jul 2, 2020 · 4 comments · Fixed by #2806

HackbrettXXX commented on Jul 2, 2020 — Collaborator

When using the html method, it is possible to inject code that is executed in the user context. E.g. like this:

```
const doc = new jsPDF();
window.html2canvas = html2canvas;
const html = `
<p id='test'>a</p>
<img src=x onerror=eval("document.getElementById('test').innerHTML=window.location") />
`;
doc.html(html, {
  callback: function (doc) {
    doc.save();
  }
});
```

E.g., this line seems to be suspicious: https://github.com/MrRio/jsPDF/blob/master/src/modules/html.js#L52.

We need to analyze how to fix this and if there is other vulnerable code.

AdamGold commented on Jul 5, 2020 — Contributor

I would say there are 3 possible ways to go here:

1. Use an external library to sanitize the `html` variables. Best option in my eyes - There are a few maintained libraries for doing this (e.g https://github.com/leizongmin/js-xss) and it will sure be more secure than the next option.
2. Do the sanitization ourselves. This would require a continued maintenance of the filtering methods.
3. Mention in the docs that this function expects **already sanitized** input.

HackbrettXXX commented on Jul 6, 2020 — Collaborator Author

I think we should go with the first option. However, I think we should allow the user to disable the sanitizing in case they really want scripts to be executed. E.g.

```
doc.html(html, {
  callback: ...
  allowScriptExecution: true/false // default: false
})
```

@AdamGold could you maybe prepare a pull request? If you don't have the time for that, I'll try it myself. Since I'm no expert in this field, I would very much appreciate if you could review it.

To clarify: the `html` method is only vulnerable when the html is passed as string, right? When passing DOM elements, scripts are not executed.

AdamGold commented on Jul 9, 2020 • edited — Contributor

@HackbrettXXX
Decided to go with https://github.com/cure53/DOMPurify for a few reasons:

1. Newest version does not contain any known vulnerabilities: https://snyk.io/vuln/npm:dompurify
2. Easy to use
3. They offer a bug bounty program
4. Worked really well on the payloads that I've tried:

```
                const doc = new jsPDF();
                window.html2canvas = html2canvas;
                const html = `
<p id='test'>a</p>
        <img src=x onerror=eval("document.getElementById('test').innerHTML=window.location") />
<img src=x onerror=alert('XSS');>
<img src=x onerror=alert('XSS')//
<img src=x onerror=alert(String.fromCharCode(88,83,83));>
<img src=x oneonerrorrror=alert(String.fromCharCode(88,83,83));>
<img src=x:alert(alt) onerror=eval(src) alt=xss>
"><img src=x onerror=alert('XSS');>
"><img src=x onerror=alert(String.fromCharCode(88,83,83));>
<svgonload=alert(1)>
<svg/onload=alert('XSS')>
<svg onload=alert(1)//
<svg/onload=alert(String.fromCharCode(88,83,83))>
<svg id=alert(1) onload=eval(id)>
"><svg/onload=alert(String.fromCharCode(88,83,83))>
"><svg/onload=alert(/XSS/)
<sCrIpt>alert(1)</ScRipt>
        `;
        doc.html(html, {
        callback: function (doc) {
        }
        });
```

AdamGold mentioned this issue on Jul 9, 2020

feat: 🎸 sanitize HTML in createElement #2806

Merged

HackbrettXXX commented on Jul 9, 2020 Collaborator Author

The vulnerability on synk for documentation: https://snyk.io/vuln/SNYK-JS-JSPDF-575256

HackbrettXXX closed this as completed in #2806 on Jul 9, 2020

---

ikornienko mentioned this issue on Aug 19, 2020

Address CVE-2020-7690 #2862

Closed

SirJalias mentioned this issue on Oct 20, 2020

Address CVE-2020-7691 (very similar to CVE-2020-7690) #2971

Closed

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Successfully merging a pull request may close this issue.

feat: 🎸 sanitize HTML in createElement
AdamGold/jsPDF

2 participants