JSDom Improper Loading of Local Resources

Medium

← View More Research Advisories

Synopsis

JSDom improperly allows the loading of local resources. Modern browser best practices dictate that the loading of local resources should be disallowed by default.

From documentation, JSDom does not, by default, load any subresources. Users must enable the loading of resources/subresources. For example, when creating a new JSDOM object, the resources item can be set to "usable" to allow the loading of external resources:

```
const dom = new JSDOM(source, {url: "http://localhost:8080/", resources: "usable"});
```

The issue here is that this setting also enables the loading of local resources. For example, the following code snippet verifies that JSDOM is attempting to access a local resource by using a non-existent file to throw an error:

```
const jsdom = require("jsdom");
const { JSDOM } = jsdom;
source = '<iframe src="file:///does_not_exist" />';
const dom = new JSDOM(source, {url: "http://localhost:8080/", resources: "usable"});
console.log(dom.window.document.body.parentElement.outerHTML);
```

Output when running the above:

```
~/r/b/u/pocs ❯❯❯ node -v; npm -v; npm list | grep jsdom;
 v15.8.0
 7.5.1
 pocs@ /Users/user/research/browser/jsdom/pocs
 └── jsdom@16.4.0
 ~/r/b/j/pocs ❯❯❯ node local_resources.js
 <html><head></head><body><iframe src="file:///does_not_exist"></iframe></body></html>
 Error: Could not load iframe: "file:///does_not_exist"
 at onErrorWrapped (/Users/user/research/browser/jsdom/pocs/node_modules/jsdom/lib/jsdom/browser/resources/per-document-resource-loader.js:38:19)
 at Object.check (/Users/user/research/browser/jsdom/pocs/node_modules/jsdom/lib/jsdom/browser/resources/resource-queue.js:72:23)
 at /Users/user/research/browser/jsdom/pocs/node_modules/jsdom/lib/jsdom/browser/resources/resource-queue.js:124:14
 at processTicksAndRejections (node:internal/process/task_queues:94:5) [Error: ENOENT: no such file or directory, open '/does_not_exist']
{ errno: -2, code: 'ENOENT', syscall: 'open', path: '/does_not_exist' }
```

While this issue alone is not terribly severe, when paired with having scripts enabled (runScripts parameter being set to "dangerously"), this could allow these local resources to be acted upon in malicious ways, such as exfiltrating sensitive information. We understand that the documentation warns against enabling this feature on untrusted input, but many downstream libraries and applications depend heavily on this feature (such as zombie.js). In other substantially more complex applications, it's possible that other exfiltration vectors could be possible without the addition of enabling scripts, such as side channel attacks via CSS requests. We understand this latter scenario is highly unlikely, but bring it up to illustrate that simply leaving scripts disabled is not necessarily a mitigation.

We highly recommend adding an additional parameter such as "localResources" to enable this functionality alongside the existing "resources" parameter. For example, to enable local resources in Google chrome, it must be launched using the "--allow-file-access-from-files" flag.

Note: The maintainer(s) of JSDom dispute this finding and disagree that it is a security concern. From the maintainer: " When you explicitly opt in to loading of data via resources: "usable", you are opting in to exfiltration. And when you explicitly opt in to script execution with runScripts: "dangerously", you are opting in to running arbitrary Node.js code on your machine. "

Solution

An update to resolve this issue has not been introduced into JSDom at the time of writing. In order to mitigate the issue, Tenable and the JSDom maintainers recommend disabling resource loading or running your applications only against known and trusted inputs. Tenable has suggested to the JSDom maintainers to add an additional flag ("localResources") alongside the existing "resources" flag that would control the loading of local resources.

Disclosure Timeline

February 5, 2020 - Tenable discovers issue.
February 10, 2020 - Tenable discloses to Tidelift (disclosure intermediary for JSDom).
February 10, 2020 - Tidelift acknowledges.
February 11, 2020 - JSDom maintainer disputes finding.
February 11, 2020 - Tenable requests clarification on testing scenario.
February 12, 2020 - Tidelift passes request along.
February 12, 2020 - Maintainer and Tenable disagree on security implications.
February 12, 2020 - Maintainer confirms that updates will not be made to the library.

Risk Information

CVE ID: CVE-2021-20066
Tenable Advisory ID: TRA-2021-05
CVSSv3 Base / Temporal Score: 5.6 / 5.3
CVSSv3 Vector: AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:L/A:L

February 16, 2021 - Initial release