ES6: NEXT GEN JS

Peoria JS and Web Professionals

July, 2018

SANITIZE USER INPUT

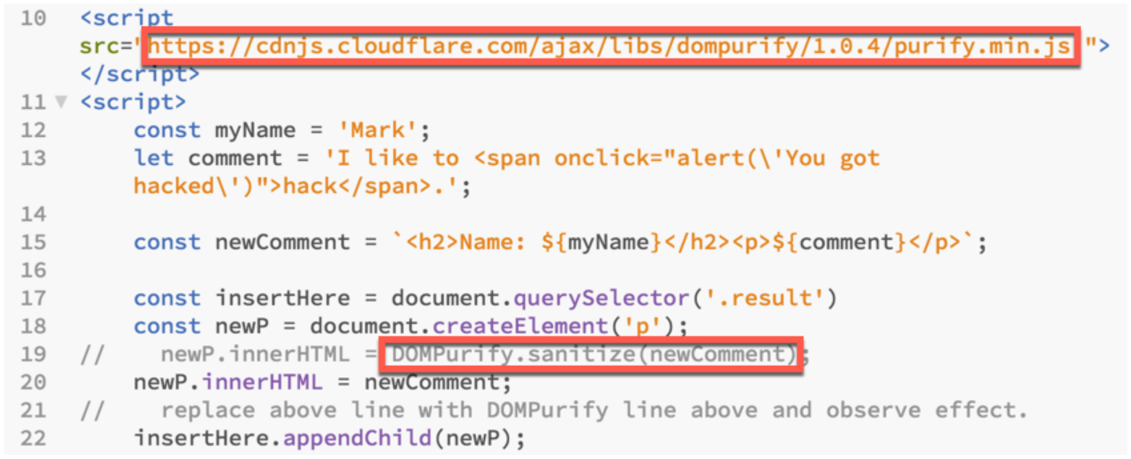
XSS is a real security exposure

Reference: https://cdnjs.com/libraries/dompurify

Note: If you allow anyone to upload data to your site, it is always a good idea to sanitize their input. Trust your visitors, never trust their input. We can use a minified version of dompurify along with tagged templates to sanitize user input. There are other approaches. It doesn't matter which one you use; just use one.

Minified library to include (this is also in the example):
https://cdnjs.cloudflare.com/ajax/libs/dompurify/1.0.4/purify.min.js

Code snippet:



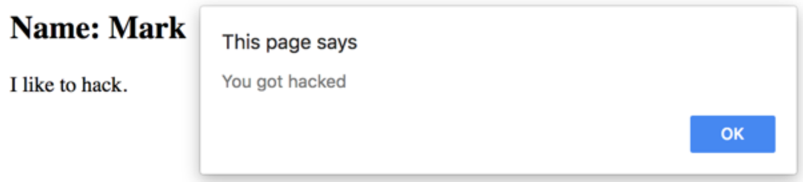
```
<script
src="https://cdnjs.cloudflare.com/ajax/libs/dompurify/1.0.4/purify.min.js">
</script>
<script>
    const myName = 'Mark';
    let comment = 'I like to <span onclick="alert(\'You got
    hacked\')">hack</span>.';

    const newComment = `<h2>Name: ${myName}</h2><p>${comment}</p>`;

    const insertHere = document.querySelector('.result')
    const newP = document.createElement('p');
//    newP.innerHTML = DOMPurify.sanitize(newComment);
    newP.innerHTML = newComment;
//    replace above line with DOMPurify line above and observe effect.
    insertHere.appendChild(newP);
```

Note that the DOMPurify code is included, but not called.

Result when viewed in the browser is shown below. This is not good. It could be any sort of attack not something as simple as this.


Name: Mark

I like to hack.

This page says

You got hacked

I strongly encourage you to always sanitize any user input. Yes, this contrived example is hard coded, but it is quite easy to accept data from a form field and pass that to a server.

Reference file: **02Security.html**