Security

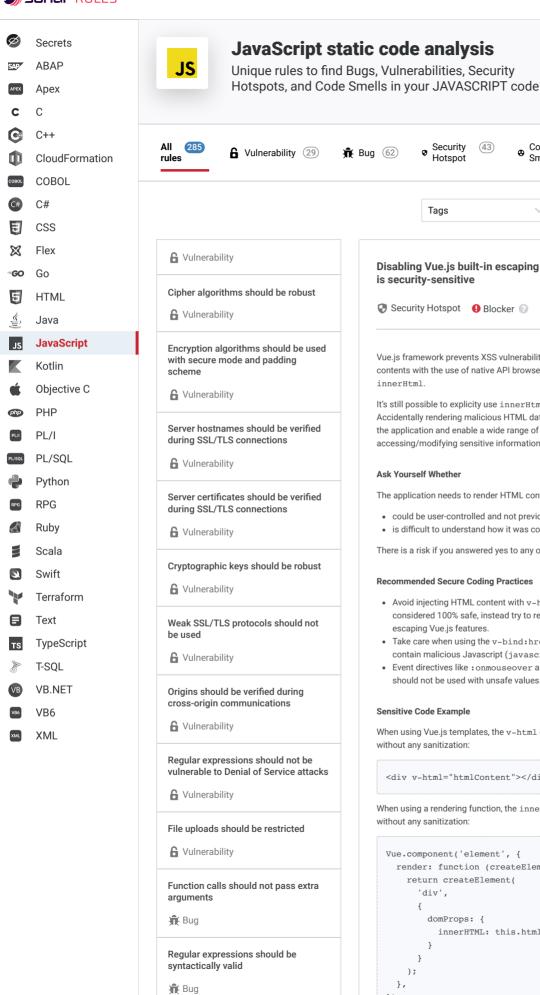
Hotspot

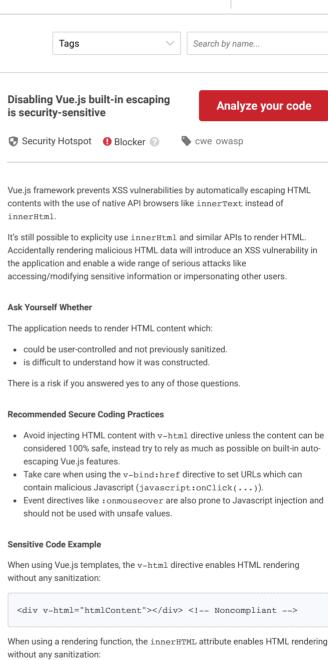
(43)



Products >

O Quick 41 Fix





render: function (createElement) { return createElement(

innerHTML: this.htmlContent, // Noncompliant

'div',

domProps: {

{

); },

});

Getters and setters should access the

expected fields

R Bug

"super()" should be invoked appropriately

R Bug

"Symbol" should not be used as a constructor

👬 Bug

Results of "in" and "instanceof" should be negated rather than operands

<table-of-contents> Bug

When using JSX, the domPropsInnerHTML attribute enables HTML rendering without any sanitization:

<div domPropsInnerHTML={this.htmlContent}></div> <!-- Noncom</pre>

Compliant Solution

When using Vue.js templates, putting the content as a child node of the element is safe:

```
<div>{{ htmlContent }}</div>
```

When using a rendering function, using the innerText attribute or putting the content as a child node of the element is safe:

```
Vue.component('element', {
  render: function (createElement) {
    return createElement(
        'div',
        {
        domProps: {
            innerText: this.htmlContent,
        }
     },
     this.htmlContent // Child node
    );
},
});
```

When using JSX, putting the content as a child node of the element is safe:

```
<div>{this.htmlContent}</div>
```

See

- OWASP Top 10 2021 Category A3 Injection
- OWASP Top 10 2017 Category A7 Cross-Site Scripting (XSS)
- <u>MITRE, CWE-79</u> Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
- Vue.js Security Injecting HTML

Available In:

sonarcloud 🔂 | sonarqube

© 2008-2022 SonarSource S.A., Switzerland. All content is copyright protected. SONAR, SONARSOURCE, SONARLINT, SONARQUBE and SONARCLOUD are trademarks of SonarSource S.A. All other trademarks and copyrights are the property of their respective owners. All rights are expressly reserved.

Privacy Policy