Secrets
ABAP
Apex
C
C++
CloudFormation
COBOL
C#
CSS
Flex
Go
HTML
Java
JavaScript
Kotlin
Objective C
PHP
PL/I
PL/SQL
Python
RPG
Ruby
Scala
Swift
Terraform
Text
TypeScript
T-SQL
VB.NET
VB6
XML

TypeScript static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your TYPESCRIPT code

All rules 279 | Vulnerability 27 | Bug 51 | Security Hotspot 43 | Code Smell 158 | Quick Fix 50

Tags | Search by name...

added
Code Smell

Primitive types should be omitted from initialized or defaulted declarations
Code Smell

Non-null assertions should not be used
Code Smell

"undefined" should not be assigned
Code Smell

Trailing commas should not be used
Code Smell

Array constructors should not be used
Code Smell

Quotes for string literals should be used consistently
Code Smell

Statements should end with semicolons
Code Smell

Comments should not be located at the end of lines of code
Code Smell

Loops should not contain more than a single "break" or "continue" statement
Code Smell

Variable, property and parameter names should comply with a naming convention
Code Smell

Lines should not end with trailing whitespaces
Code Smell

Disabling strict HTTP no-referrer policy is security-sensitive

Analyze your code

Security Hotspot Minor cwe express.js owasp

HTTP header referer contains a URL set by web browsers and used by applications to track from where the user came from, it's for instance a relevant value for web analytic services, but it can cause serious privacy and security problems if the URL contains confidential information. Note that Firefox for instance, to prevent data leaks, removes path information in the Referer header while browsing privately.

Suppose an e-commerce website asks the user his credit card number to purchase a product:

```
<html>
<body>
<form action="/valid_order" method="GET">
Type your credit card number to purchase products:
<input type=text id="cc" value="1111-2222-3333-4444">
<input type=submit>
</form>
</body>
```

When submitting the above HTML form, a HTTP GET request will be performed, the URL requested will be https://example.com/valid_order?cc=1111-2222-3333-4444 with credit card number inside and it's obviously not secure for these reasons:

- URLs are stored in the history of browsers.
- URLs could be accidentally shared when doing copy/paste actions.
- URLs can be stolen if a malicious person looks at the computer screen of an user.

In addition to these threats, when further requests will be performed from the "valid_order" page with a simple legitimate embedded script like that:

```
<script src="https://webanalyticservices_example.com/track">
```

The referer header which contains confidential information will be send to a third party web analytic service and cause privacy issue:

```
GET /track HTTP/2.0
Host: webanalyticservices_example.com
Referer: https://example.com/valid_order?cc=1111-2222-3333-4
```

Ask Yourself Whether

- Confidential information exists in URLs.
- Semantic of HTTP methods is not respected (eg: use of a GET method instead of POST when the state of the application is changed).

There is a risk if you answered yes to any of those questions.

Recommended Secure Coding Practices

Files should contain an empty newline at the end

Code Smell

An open curly brace should be located at the end of a line

Code Smell

Tabulation characters should not be used

Code Smell

Function and method names should comply with a naming convention

Code Smell

Confidential information should not be set inside URLs (GET requests) of the application and a safe (ie: different from `unsafe-url` or `no-referrer-when-downgrade`) referrer-Policy header, to control how much information is included in the referer header, should be used.

Sensitive Code Example

In Express.js application the code is sensitive if the helmet `referrerPolicy` middleware is disabled or used with `no-referrer-when-downgrade` or `unsafe-url`:

```
const express = require('express');
const helmet = require('helmet');

app.use(
  helmet.referrerPolicy({
    policy: 'no-referrer-when-downgrade' // Sensitive: no-re
  })
);
```

Compliant Solution

In Express.js application a secure solution is to user the helmet referrer policy middleware set to `no-referrer`:

```
const express = require('express');
const helmet = require('helmet');

let app = express();

app.use(
  helmet.referrerPolicy({
    policy: 'no-referrer' // Compliant
  })
);
```

See

- OWASP Top 10 2021 Category A1 - Broken Access Control
- OWASP Top 10 2017 Category A3 - Sensitive Data Exposure
- developer.mozilla.org - Referrer-Policy
- developer.mozilla.org - Referer header: privacy and security concerns
- MITRE, CWE-200 - Exposure of Sensitive Information to an Unauthorized Actor

Available In:

sonarcloud sonarqube