




 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... 🔍

 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

### Writing cookies is security-sensitive

Security Hotspot  Minor ?

Using cookies is security-sensitive. It has led in the past to the following vulnerabilities:

- [CVE-2018-11639](#)
- [CVE-2016-6537](#)

Attackers can use widely-available tools to read cookies. Any sensitive information they may contain will be exposed.

This rule flags code that writes cookies.

#### Ask Yourself Whether

- sensitive information is stored inside the cookie.

You are at risk if you answered yes to this question.

#### Recommended Secure Coding Practices

Cookies should only be used to manage the user session. The best practice is to keep all user-related information server-side and link them to the user session, never sending them to the client. In a very few corner cases, cookies can be used for non-sensitive information that need to live longer than the user session.

Do not try to encode sensitive information in a non human-readable format before writing them in a cookie. The encoding can be reverted and the original information will be exposed.

Using cookies only for session IDs doesn't make them secure. Follow [OWASP best practices](#) when you configure your cookies.

As a side note, every information read from a cookie should be [Sanitized](#).

#### Sensitive Code Example

```
// === Built-in NodeJS modules ===
const http = require('http');
const https = require('https');

http.createServer(function(req, res) {
  res.setHeader('Set-Cookie', ['type=ninja', 'lang=js']); //
});
https.createServer(function(req, res) {
  res.setHeader('Set-Cookie', ['type=ninja', 'lang=js']); //
});

// === ExpressJS ===
const express = require('express');
const app = express();
app.use(function(req, res, next) {
  res.cookie('name', 'John'); // Sensitive
});
```

https://rules.sonarsource.com/typescript/RSPEC-2255

1/2

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

```
// === In browser ===  
// Set cookie  
document.cookie = "name=John"; // Sensitive
```

See

- [OWASP Top 10 2017 Category A3](#) - Sensitive Data Exposure
- [MITRE, CWE-312](#) - Cleartext Storage of Sensitive Information
- [MITRE, CWE-315](#) - Cleartext Storage of Sensitive Information in a Cookie
- Derived from FindSecBugs rule [COOKIE\\_USAGE](#)

Deprecated

This rule is deprecated, and will eventually be removed.

Available In:  
 