




 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... 🔍

 Vulnerability
Repeated patterns in regular expressions should not match the empty string
 Bug
Empty collections should not be accessed or iterated
 Bug
"delete" should be used only with object properties
 Bug
Function parameters, caught exceptions and foreach variables' initial values should not be ignored
 Bug
Forwarding client IP address is security-sensitive
 Security Hotspot
Allowing confidential information to be logged is security-sensitive
 Security Hotspot
Allowing browsers to perform DNS prefetching is security-sensitive
 Security Hotspot
Disabling Certificate Transparency monitoring is security-sensitive
 Security Hotspot
Disabling Strict-Transport-Security policy is security-sensitive
 Security Hotspot
Disabling strict HTTP no-referrer policy is security-sensitive
 Security Hotspot
Allowing browsers to sniff MIME types is security-sensitive

"NaN" should not be used in comparisons

Analyze your code

 Bug

 Major

 Quick Fix

NaN is not equal to anything, even itself. Testing for equality or inequality against NaN will yield predictable results, but probably not the ones you want.

Instead, the best way to see whether a variable is equal to NaN is to use `Number.isNaN()`, since ES2015, or (perhaps counter-intuitively) to compare it to itself. Since `NaN !== NaN`, when `a !== a`, you know it must equal NaN.

Noncompliant Code Example

```
var a = NaN;

if (a === NaN) { // Noncompliant; always false
  console.log("a is not a number"); // this is dead code
}

if (a !== NaN) { // Noncompliant; always true
  console.log("a is not NaN"); // this statement is not needed
}
```

Compliant Solution





```
if (Number.isNaN(a)) {
  console.log("a is not a number");
}

if (!Number.isNaN(a)) {
  console.log("a is not NaN");
}
```

Available In:

 |  | 

© 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](#)

 Security Hotspot
<div>Disabling content security policy frame-ancestors directive is security-sensitive</div> <div> Security Hotspot</div>
<div>Allowing mixed-content is security-sensitive</div> <div> Security Hotspot</div>
<div>Disabling content security policy fetch directives is security-sensitive</div> <div> Security Hotspot</div>
<div>Disabling resource integrity features is security-sensitive</div>