




 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 rules279

 Vulnerability27

 Bug51


 Security Hotspot43

 Code Smell158


 Quick Fix50

Tags ▾


Search by name... 🔍

 Code Smell


Primitive types should be omitted from initialized or defaulted declarations




Non-null assertions should not be used




"undefined" should not be assigned




Trailing commas should not be used




Array constructors should not be used




Quotes for string literals should be used consistently




Statements should end with semicolons




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




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






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



Lines should not end with trailing whitespaces



Disabling Certificate Transparency monitoring is security-sensitive

 Security Hotspot Minor ? cwe express.js owasp

Certificate Transparency (CT) is an open-framework to protect against identity theft when certificates are issued. Certificate Authorities (CA) electronically sign certificate after verifying the identify of the certificate owner. Attackers use, among other things, social engineering attacks to trick a CA to correctly verifying a spoofed identity/forged certificate.

CAs implement Certificate Transparency framework to publicly log the records of newly issued certificates, allowing the public and in particular the identity owner to monitor these logs to verify that his identify was not usurped.

Ask Yourself Whether

- The website identity is valuable and well-known, therefore prone to theft.

There is a risk if you answered yes to this question.

Recommended Secure Coding Practices

Implement Expect-CT HTTP header which instructs the web browser to check public CT logs in order to verify if the website appears inside and if it is not, the browser will block the request and display a warning to the user.

Sensitive Code Example

In Express.js application the code is sensitive if the expect-ct middleware is disabled:

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

let app = express();

app.use(
  helmet({
    expectCt: false // Sensitive
  })
);
```

Compliant Solution

In Express.js application the expect-ct middleware is the standard way to implement expect-ct. Usually, the deployment of this policy starts with the report only mode (`enforce: false`) and with a low `maxAge` (the number of seconds the policy will apply) value and next if everything works well it is recommended to block future connections that violate Expect-CT policy (`enforce: true`) and greater value for `maxAge` directive:

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

https://rules.sonarsource.com/typescript/RSPEC-5742

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

```
let app = express();

app.use(helmet.expectCt({
  enforce: true,
  maxAge: 86400
})); // Compliant
```

See

- [OWASP Top 10 2021 Category A5](#) - Security Misconfiguration
- [OWASP Top 10 2017 Category A3](#) - Sensitive Data Exposure
- [developer.mozilla.org](#) - Certificate Transparency
- [wikipedia.org](#) - Certificate Authority

Available In:

