




 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

promises

Code Smell

A conditionally executed single line should be denoted by indentation

Code Smell

Conditionals should start on new lines

Code Smell

Cognitive Complexity of functions should not be too high

Code Smell

"void" should not be used

Code Smell

Loop counters should not be assigned to from within the loop body

Code Smell

"for" loop increment clauses should modify the loops' counters

Code Smell

Functions should not be empty

Code Smell

Server-side requests should not be vulnerable to forging attacks

Vulnerability

Non-empty statements should change control flow or have at least one side-effect

Bug

Regular expressions with the global flag should be used with caution

Bug

Replacement strings should reference existing regular expression groups

Bug

Tags ▾

Search by name... 🔍

Weak SSL/TLS protocols should not be used

Analyze your code

Vulnerability

Critical

cwe privacy owasp sans-top25

This rule raises an issue when an insecure TLS protocol version (i.e. a protocol different from "TLSv1.2", "TLSv1.3", "DTLSv1.2", or "DTLSv1.3") is used or allowed.

It is recommended to enforce TLS 1.2 as the minimum protocol version and to disallow older versions like TLS 1.0. Failure to do so could open the door to downgrade attacks: a malicious actor who is able to intercept the connection could modify the requested protocol version and downgrade it to a less secure version.

Noncompliant Code Example

secureProtocol,minVersion/maxVersion and secureOptions should not be set to use weak TLS protocols (TLSv1.1 and lower):

```
let options = {
  secureProtocol: 'TLSv1_method' // Noncompliant: TLS1.0 is
};

let options = {
  minVersion: 'TLSv1.1', // Noncompliant: TLS1.1 is insecure
  maxVersion: 'TLSv1.2'
};

let options = {
  secureOptions: constants.SSL_OP_NO_SSLv2 | constants.SSL_OP_NO_TLSv1_1,
}; // Noncompliant TLS 1.1 (constants.SSL_OP_NO_TLSv1_1) is
```

https built-in module:

```
let req = https.request(options, (res) => {
  res.on('data', (d) => {
    process.stdout.write(d);
  });
}); // Noncompliant
```

tls built-in module:

```
let socket = tls.connect(443, "www.example.com", options, ()
```

request module:




```
let socket = request.get(options);
```

Compliant Solution

Set either secureProtocol or secureOptions or minVersion to use secure protocols only (TLSv1.2 and higher):

https://rules.sonarsource.com/typescript/RSPEC-4423

1/2

<div>Regular expressions should not contain control characters</div> <div> Bug</div>
<div>Alternation in regular expressions should not contain empty alternatives</div> <div> Bug</div>
<div>Mocha timeout should be disabled by setting it to "0".</div> <div> Bug</div>
<div>Unicode Grapheme Clusters should be avoided inside regex character classes</div>

```
let options = {
  secureProtocol: 'TLSv1_2_method'
};
// or
let options = {
  secureOptions: constants.SSL_OP_NO_SSLv2 | constants.SSL_O
};
// or
let options = {
  minVersion: 'TLSv1.2'
};
```

[https](#) built-in module:

```
let req = https.request(options, (res) => {
  res.on('data', (d) => {
    process.stdout.write(d);
  });
}); // Compliant
```

[tls](#) built-in module:

```
let socket = tls.connect(443, "www.example.com", options, ()
```

[request](#) module:

```
let socket = request.get(options);
```

See

- [OWASP Top 10 2021 Category A2](#) - Cryptographic Failures
- [OWASP Top 10 2021 Category A7](#) - Identification and Authentication Failures
- [OWASP Top 10 2017 Category A3](#) - Sensitive Data Exposure
- [OWASP Top 10 2017 Category A6](#) - Security Misconfiguration
- [Mobile AppSec Verification Standard](#) - Network Communication Requirements
- [OWASP Mobile Top 10 2016 Category M3](#) - Insecure Communication
- [MITRE, CWE-327](#) - Inadequate Encryption Strength
- [MITRE, CWE-326](#) - Use of a Broken or Risky Cryptographic Algorithm
- [SANS Top 25](#) - Porous Defenses
- [SSL and TLS Deployment Best Practices - Use secure protocols](#)

Available In:

