




 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



JavaScript static code analysis

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

All rules285

Vulnerability29

Bug62

Security Hotspot43


Code Smell151

Quick Fix41


Tags ▾

Search by name... 🔍


redundant




Default export names and file names should match




The global "this" object should not be used




"catch" clauses should do more than rethrow




Boolean checks should not be inverted




Deprecated APIs should not be used




Wrapper objects should not be used for primitive types




Multiline string literals should not be used




Local variables should not be declared and then immediately returned or thrown




Unused local variables and functions should be removed



Function call arguments should not start on new lines



"switch" statements should have at least 3 "case" clauses



Allowing requests with excessive content length is security-sensitive

Analyze your code

Security Hotspot

Major

cwe owasp express.js

Rejecting requests with significant content length is a good practice to control the network traffic intensity and thus resource consumption in order to prevents DoS attacks.

Ask Yourself Whether

- size limits are not defined for the different resources of the web application.
- the web application is not protected by **rate limiting** features.
- the web application infrastructure has limited resources.

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

Recommended Secure Coding Practices

- For most of the features of an application, it is recommended to limit the size of requests to:
 - lower or equal to 8mb for file uploads.
 - lower or equal to 2mb for other requests.

It is recommended to customize the rule with the limit values that correspond to the web application.

Sensitive Code Example

formidable file upload module:

```
const form = new Formidable();
form.maxFileSize = 10000000; // Sensitive: 10MB is more than

const formDefault = new Formidable(); // Sensitive, the default
```

multer (Express.js middleware) file upload module:

```
let diskUpload = multer({
  storage: diskStorage,
  limits: {
    fileSize: 10000000; // Sensitive: 10MB is more than the
  }
});

let diskUploadUnlimited = multer({ // Sensitive: the default
  storage: diskStorage,
});
```

body-parser module:

```
// 4MB is more than the recommended limit of 2MB for non-file
let jsonParser = bodyParser.json({ limit: "4mb" }); // Sensitive
let urlencodedParser = bodyParser.urlencoded({ extended: false });
```

https://rules.sonarsource.com/javascript/RSPEC-5693

1/2

A "while" loop should be used instead of a "for" loop

 Code Smell

Unnecessary imports should be removed

 Code Smell

Return of boolean expressions should not be wrapped into an "if-then-else" statement

 Code Smell

Boolean literals should not be used in comparisons

Compliant Solution

[formidable](#) file upload module:

```
const form = new Formidable();
form.maxFileSize = 8000000; // Compliant: 8MB
```

[multer](#) (Express.js middleware) file upload module:

```
let diskUpload = multer({
  storage: diskStorage,
  limits: {
    fileSize: 8000000 // Compliant: 8MB
  }
});
```

[body-parser](#) module:

```
let jsonParser = bodyParser.json(); // Compliant, when the 1
let urlencodedParser = bodyParser.urlencoded({ extended: fal
```

See

- [OWASP Top 10 2021 Category A5](#) - Security Misconfiguration
- [Owasp Cheat Sheet](#) - Owasp Denial of Service Cheat Sheet
- [OWASP Top 10 2017 Category A6](#) - Security Misconfiguration
- [MITRE, CWE-770](#) - Allocation of Resources Without Limits or Throttling
- [MITRE, CWE-400](#) - Uncontrolled Resource Consumption

Available In:

sonarcloud  **sonarqube** 