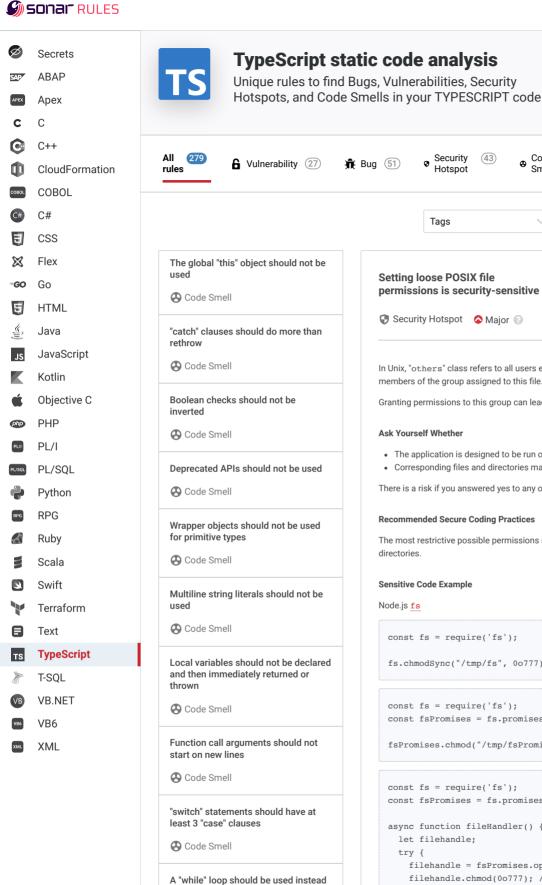


Products ~

O Quick 50 Fix

Q



```
Setting loose POSIX file
                                             Analyze your code
permissions is security-sensitive
cwe sans-top25 owasp
In Unix, "others" class refers to all users except the owner of the file and the
members of the group assigned to this file.
Granting permissions to this group can lead to unintended access to files.
 • The application is designed to be run on a multi-user environment.
 • Corresponding files and directories may contain confidential information.
There is a risk if you answered yes to any of those questions.
Recommended Secure Coding Practices
The most restrictive possible permissions should be assigned to files and
 const fs = require('fs');
  fs.chmodSync("/tmp/fs", 0o777); // Sensitive
 const fs = require('fs');
 const fsPromises = fs.promises;
  fsPromises.chmod("/tmp/fsPromises", 0o777); // Sensitive
 const fs = require('fs');
 const fsPromises = fs.promises
 async function fileHandler() {
      filehandle = fsPromises.open('/tmp/fsPromises', 'r');
      filehandle.chmod(00777); // Sensitive
    } finally {
      if (filehandle !== undefined)
        filehandle.close();
 }
Node.js process.umask
 const process = require('process');
```

(158)

Search by name.

of a "for" loop

Code Smell

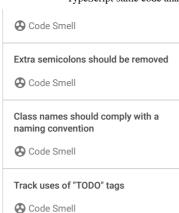
Code Smell

comparisons

removed

Unnecessary imports should be

Boolean literals should not be used in



Web SOL databases should not be

used

❸ Vulnerability

```
process.umask(00000); // Sensitive
```

Compliant Solution

Node.js <u>fs</u>

```
const fs = require('fs');
fs.chmodSync("/tmp/fs", 00770); // Compliant
```

```
const fs = require('fs');
const fsPromises = fs.promises;
fsPromises.chmod("/tmp/fsPromises", 0o770); // Compliant
```

```
const fs = require('fs');
const fsPromises = fs.promises

async function fileHandler() {
  let filehandle;
  try {
    filehandle = fsPromises.open('/tmp/fsPromises', 'r');
    filehandle.chmod(00770); // Compliant
  } finally {
    if (filehandle !== undefined)
      filehandle.close();
  }
}
```

Node.js process.umask

```
const process = require('process');
process.umask(00007); // Compliant
```

See

- OWASP Top 10 2021 Category A1 Broken Access Control
- OWASP Top 10 2021 Category A4 Insecure Design
- OWASP Top 10 2017 Category A5 Broken Access Control
- OWASP File Permission
- MITRE, CWE-732 Incorrect Permission Assignment for Critical Resource
- MITRE, CWE-266 Incorrect Privilege Assignment
- SANS Top 25 Porous Defenses

Available In:

sonarcloud 🐼 | sonarqube

© 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