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

C# static code analysis

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

All rules 409 | Vulnerability 34 | Bug 76 | Security Hotspot 28 | Code Smell 271 | Quick Fix 52

Tags

Search by name...

commands from user input is security-sensitive
Security Hotspot

Deserializing objects without performing data validation is security-sensitive
Security Hotspot

Disabling ASP.NET "Request Validation" feature is security-sensitive
Security Hotspot

Allowing requests with excessive content length is security-sensitive
Security Hotspot

Setting loose file permissions is security-sensitive
Security Hotspot

Formatting SQL queries is security-sensitive
Security Hotspot

Using hardcoded IP addresses is security-sensitive
Security Hotspot

"goto" statement should not be used
Code Smell

"new Guid()" should not be used
Code Smell

Parameter validation in "async"/"await" methods should be wrapped
Code Smell

Parameter validation in yielding methods should be wrapped
Code Smell

Events should have proper arguments

Non-flags enums should not be used in bitwise operations

Analyze your code

Code Smell | Critical | Quick Fix | convention

`enums` are usually used to identify distinct elements in a set of values. However `enums` can be treated as bit fields and bitwise operations can be used on them to combine the values. This is a good way of specifying multiple elements of set with a single value. When `enums` are used this way, it is a best practice to mark the `enum` with the `FlagsAttribute`.

Noncompliant Code Example

```
enum Permissions
{
  None = 0,
  Read = 1,
  Write = 2,
  Execute = 4
}
// ...

var x = Permissions.Read | Permissions.Write;  // Noncomplia
```

Compliant Solution

```
[Flags]
enum Permissions
{
  None = 0,
  Read = 1,
  Write = 2,
  Execute = 4
}
// ...

var x = Permissions.Read | Permissions.Write;
```

Available In:

sonarlint | sonarcloud | sonarqube

Code Smell

"P/Invoke" methods should not be visible

Code Smell

Native methods should be wrapped

Code Smell

Methods should not have identical implementations

Code Smell

Non-flags enums should not be marked with "FlagsAttribute"

Code Smell