



Using hardcoded IP addresses is

"goto" statement should not be used

"new Guid()" should not be used

Parameter validation in yielding

Events should have proper arguments

"P/Invoke" methods should not be

Native methods should be wrapped

Methods should not have identical

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

methods should be wrapped

security-sensitive

Code Smell

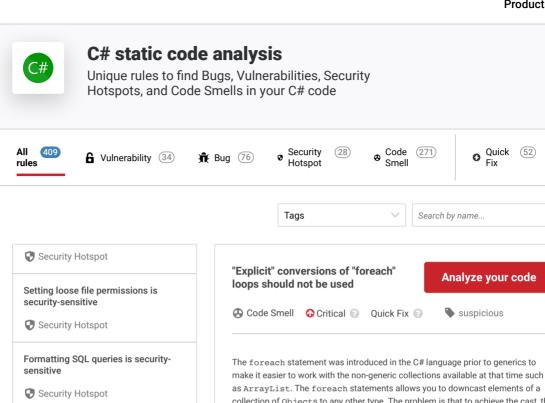
implementations

Code Smell

wrapped

Parameter validation in "async"/"await" methods should be

Security Hotspot



collection of Objects to any other type. The problem is that to achieve the cast, the foreach statements silently performs explicit type conversion, which at runtime can result in an InvalidCastException.

C# code iterating on generic collections or arrays should not rely on foreach statement's silent explicit conversions.

Noncompliant Code Example

```
public class Fruit { }
public class Orange : Fruit { }
public class Apple : Fruit { }
class MyTest
{
  public void Test()
    var fruitBasket = new List<Fruit>();
    fruitBasket.Add(new Orange());
    fruitBasket.Add(new Orange());
    // fruitBasket.Add(new Apple()); // uncommenting this 1
    foreach (Fruit fruit in fruitBasket)
      var orange = (Orange)fruit; // This "explicit" convers
    foreach (Orange orange in fruitBasket) // Noncompliant
    {
```

Compliant Solution

```
var fruitBasket = new List<Orange>();
fruitBasket.Add(new Orange());
fruitBasket.Add(new Orange());
// fruitBasket.Add(new Apple()); // uncommenting this line
foreach (Orange orange in fruitBasket)
{
```

Code Smell

Classes implementing "IEquatable<T>" should be sealed

Code Smell

"GC.SuppressFinalize" should not be called

Code Smell

Objects should not be disposed more than once

Code Smell

Parameter names used into ArgumentException constructors or

```
var fruitBasket = new List<Fruit>();
fruitBasket.Add(new Orange());
fruitBasket.Add(new Orange());
fruitBasket.Add(new Apple());

foreach (Orange orange in fruitBasket.OfType<Orange>())
{
    ...
}
```

Exceptions

The rule ignores iterations on collections of objects. This includes legacy code that uses ArrayList. Furthermore, the rule does not report on cases when user defined conversions are being called.

Available In:

sonarlint ⊕ | 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