C# static code analysis

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

Properties should not make collection or array copies

Analyze your code

Code Smell Critical api-design performance

Most developers expect property access to be as efficient as field access. However, if a property returns a copy of an array or collection, it will be much slower than simple field access, contrary to the caller’s likely expectations. Therefore, such properties should be refactored into methods so that callers are not surprised by the unexpectedly poor performance.

This rule detects calls to `ToList`, `ToArray` and array `Clone`.

Noncompliant Code Example

```
private List<string> _foo = new List<string> { "a", "b", "c"
public IEnumerable<string> Foo  // Noncompliant
{
    get
    {
        return _foo.ToList();
    }
}

private string[] _bar = new string[] { "a", "b", "c" };
public IEnumerable<string> Bar // Noncompliant
{
    get
    {
        return (string[])_bar.Clone();
    }
}
```

Compliant Solution

```
private List<string> _foo = new List<string> { "a", "b", "c"
private string[] _bar = new string[] { "a", "b", "c" };

public IEnumerable<string> GetFoo()
{
    return _foo.ToList();
}

public IEnumerable<string> GetBar()
{
    return (string[])_bar.Clone();
}
```

Available In:

sonarlint sonarcloud sonarqube

Code Smell

"ServiceContract" and "OperationContract" attributes should be used together

Code Smell

Composite format strings should be used correctly

Code Smell

Exceptions should not be explicitly rethrown

Code Smell

"abstract" classes should not have