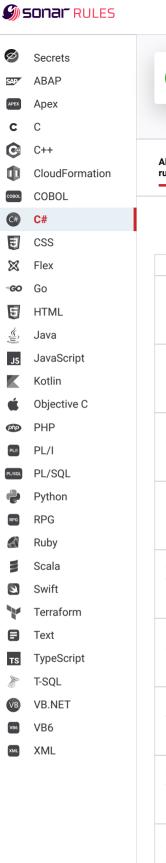
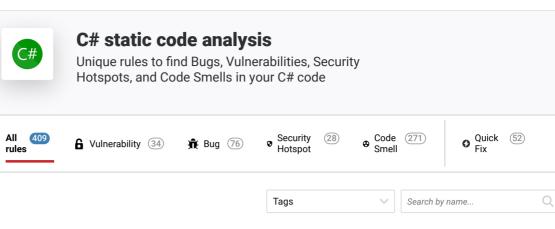
Analyze your code

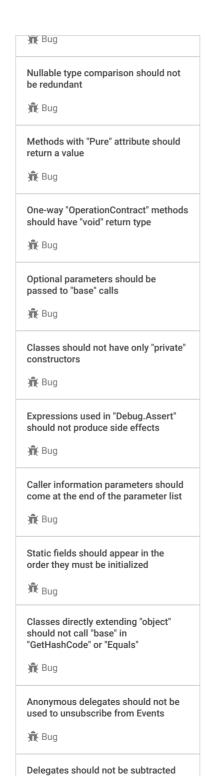






Locks should be released

Rug Oritical O



📆 Bug



cwe multi-threading

If a lock is known to be held or acquired, and then released within a method, then it

## **Compliant Solution**

```
class MyClass
 private object obj = new object();
 public void DoSomethingWithMonitor()
   lock(obj) // lock() {...} is easier to use than explicit
      if (IsInitialized())
```

```
"async" methods should not return
"void"

Rug

"ThreadStatic" should not be used on non-static fields

Rug

"IDisposables" created in a "using" statement should not be returned

Rug

"ThreadStatic" fields should not be initialized
```

```
{
    }
}

private ReaderWriterLockSlim lockObj = new ReaderWriterLoc

public void DoSomethingWithReaderWriteLockSlim()
{
    lockObj.EnterReadLock();
    try
    {
        if (IsInitialized())
        {
          }
     }
     finally
    {
        lockObj.ExitReadLock();
    }
}
```

## See

- MITRE, CWE-459 Incomplete Cleanup
- Synchronization of access to a shared resource.

Available In:

 $\mathsf{sonar}\mathsf{lint} \, \underline{\ominus} \mid \mathsf{sonar}\mathsf{cloud} \, \, \underline{\Diamond} \mid \mathsf{sonar}\mathsf{qube} \, \underline{)}$ 

© 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