Code Smell (101)

Search by name...

Security Hotspot 31

Tags

Cipher algorithms should be

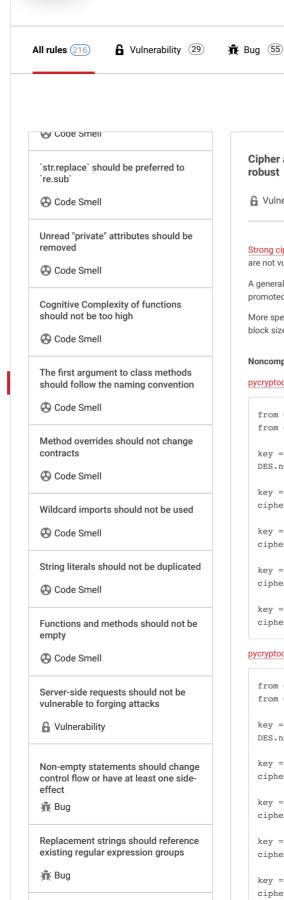




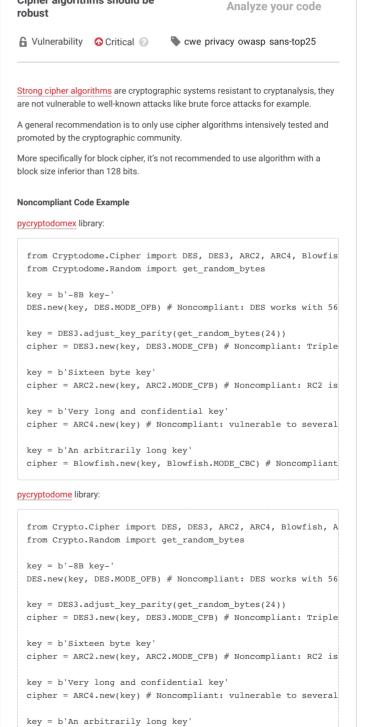


Python static code analysis

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



Alternation in regular expressions should not contain empty alternatives



cipher = Blowfish.new(key, Blowfish.MODE_CBC) # Noncompliant



Unicode Grapheme Clusters should be avoided inside regex character classes

👬 Bug

Regex alternatives should not be redundant

₩ Bug

Alternatives in regular expressions should be grouped when used with anchors



pyca library:

```
import os
from cryptography.hazmat.primitives.ciphers import Cipher, a
from cryptography.hazmat.backends import default_backend

key = os.urandom(16)
iv = os.urandom(16)

tdes4 = Cipher(algorithms.TripleDES(key), mode=None, backend
bf3 = Cipher(algorithms.Blowfish(key), mode=None, backend=de
rc42 = Cipher(algorithms.ARC4(key), mode=None, backend=defau
```

pydes library:

```
import pyDes;

des1 = pyDes.des('ChangeIt')  # Noncompliant: DES works with
des2 = pyDes.des('ChangeIt', pyDes.CBC, "\0\0\0\0\0\0\0\0\0\0",

tdes1 = pyDes.triple_des('ChangeItWithYourKey!!!!!')  # Nonc
tdes2 = pyDes.triple_des('ChangeItWithYourKey!!!!!', pyDes.C
```

pycrypto library is not maintained and therefore should not be used:

```
from Crypto.Cipher import *

des3 = DES.new('ChangeIt') # Noncompliant: DES works with 56
tdes3 = DES3.new('ChangeItChangeIt') # Noncompliant: Triple
bf2 = Blowfish.new('ChangeItWithYourKey', Blowfish.MODE_CBC,
rc21 = ARC2.new('ChangeItWithYourKey', ARC2.MODE_CFB, 'Chang
rc41 = ARC4.new('ChangeItWithYourKey') # Noncompliant: vulne
```

Compliant Solution

pycryptodomex library:

```
from Cryptodome.Cipher import AES

key = b'Sixteen byte key'
cipher = AES.new(key, AES.MODE_CCM) # Compliant
```

pycryptodome library:

```
from Crypto.Cipher import AES

key = b'Sixteen byte key'
cipher = AES.new(key, AES.MODE_CCM) # Compliant
```

pyca library:

```
import os
from cryptography.hazmat.primitives.ciphers import Cipher, a
from cryptography.hazmat.backends import default_backend

key = os.urandom(16)
iv = os.urandom(16)

aes2 = Cipher(algorithms.AES(key), modes.CBC(iv), backend=de
```

pycrypto library is not maintained and therefore should not be used:

```
from Crypto.Cipher import *
aes1 = AES.new('This is a key123', AES.MODE_CBC, 'This is an
```

See

- OWASP Top 10 2021 Category A2 Cryptographic Failures
- OWASP Top 10 2017 Category A3 Sensitive Data Exposure
- MITRE, CWE-327 Use of a Broken or Risky Cryptographic Algorithm
- SANS Top 25 Porous Defenses

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