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

Python static code analysis

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

All rules 216 | Vulnerability 29 | Bug 55 | Security Hotspot 31 | Code Smell 101

Tags

Search by name...

Replacement strings should reference existing regular expression groups
Bug

Alternation in regular expressions should not contain empty alternatives
Bug

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
Bug

New objects should not be created only to check their identity
Bug

Collection content should not be replaced unconditionally
Bug

Exceptions should not be created without being raised
Bug

Collection sizes and array length comparisons should make sense
Bug

All branches in a conditional structure should not have exactly the same implementation
Bug

The output of functions that don't return anything should not be used

Cryptographic key generation should be based on strong parameters

Analyze your code

Vulnerability | Critical | cwe privacy owasp

When generating cryptographic keys (or key pairs), it is important to use strong parameters. Key length, for instance, should provide enough entropy against brute-force attacks.

- For RSA and DSA algorithms key size should be at least 2048 bits long
- For ECC (elliptic curve cryptography) algorithms key size should be at least 224 bits long
- For RSA public key exponent should be at least 65537.

This rule raises an issue when a RSA, DSA or ECC key-pair generator is initialized using weak parameters.

It supports the following libraries:

- cryptography
- PyCrypto
- Cryptodome

Noncompliant Code Example

```
from cryptography.hazmat.primitives.asymmetric import r

dsa.generate_private_key(key_size=1024, backend=backend
rsa.generate_private_key(public_exponent=999, key_size=
ec.generate_private_key(curve=ec.SECT163R2, backend=bac
```

Compliant Solution

```
from cryptography.hazmat.primitives.asymmetric import r

dsa.generate_private_key(key_size=2048, backend=backend
rsa.generate_private_key(public_exponent=65537, key_siz
ec.generate_private_key(curve=ec.SECT409R1, backend=bac
```

See

- OWASP Top 10 2021 Category A2 - Cryptographic Failures
- OWASP Top 10 2017 Category A3 - Sensitive Data Exposure
- OWASP Top 10 2017 Category A6 - Security Misconfiguration
- ANSSI RGSv2 - Référentiel Général de Sécurité version 2
- NIST FIPS 186-4 - Digital Signature Standard (DSS)
- MITRE, CWE-326 - Inadequate Encryption Strength

Available In:

sonarlint | sonarcloud | sonarqube

Bug

"=+" should not be used instead of "+="

Bug

Increment and decrement operators should not be used

Bug

Return values from functions without side effects should not be ignored

Bug

Related "if/else if" statements should not have the same condition

Bug