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...

authenticated
Vulnerability

Cryptographic key generation should be based on strong parameters
Vulnerability

Weak SSL/TLS protocols should not be used
Vulnerability

Cipher Block Chaining IVs should be unpredictable
Vulnerability

Regular expressions should not be vulnerable to Denial of Service attacks
Vulnerability

Hashes should include an unpredictable salt
Vulnerability

Regex lookahead assertions should not be contradictory
Bug

Regex boundaries should not be used in a way that can never be matched
Bug

Exceptions' "__cause__" should be either an Exception or None
Bug

"break" and "continue" should not be used outside a loop
Bug

Break, continue and return statements should not occur in "finally" blocks
Bug

Allowing public ACLs or policies on a S3 bucket is security-sensitive

Function arguments should be passed only once

Analyze your code

Bug Blocker

When a function is called, it accepts only one value per parameter. Python interpreters will raise a SyntaxError when they see something like `myfunction(a=1, a=2)`, but there are other cases which will only fail at runtime:

- An argument is provided by value and position at the same time.
- Some arguments are provided via unpacking and the same argument is provided twice.

This rule raises an issue when a function is called with multiple values for the same parameter.

Noncompliant Code Example

```
def func(a, b, c):
    return a * b * c

func(6, 93, 31, c=62) # Noncompliant: argument "c" is d

params = {'c':31}
func(6, 93, 31, **params) # Noncompliant: argument "c"
func(6, 93, c=62, **params) # Noncompliant: argument "c
```

Compliant Solution

```
def func(a, b, c):
    return a * b * c

print(func(c=31, b=93, a=6)) # Compliant
```

Available In:

sonarlint | sonarcloud | sonarqube

Security Hotspot

Using publicly writable directories is security-sensitive

Security Hotspot

Using clear-text protocols is security-sensitive

Security Hotspot

Expanding archive files without controlling resource consumption is security-sensitive

Security Hotspot

Signalling processes is security-sensitive