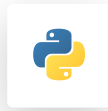


- Secrets
- ABAP
- Apex
- C
- C++
- CloudFormation
- COBOL
- C#
- CSS
- Flex
- Go
- HTML
- Java
- JavaScript
- Kotlin
- Objective C
- PHP
- PL/I
- PL/SQL
- Python**
- RPG
- Ruby
- Scala
- Swift
- Terraform
- Text
- TypeScript
- T-SQL
- VB.NET
- VB6
- XML



## 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...



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

Security Hotspot

Configuring loggers is security-sensitive

Security Hotspot

Using weak hashing algorithms is security-sensitive

Security Hotspot

Disabling CSRF protections is security-sensitive

Security Hotspot

Using non-standard cryptographic algorithms is security-sensitive

Security Hotspot

Using pseudorandom number generators (PRNGs) is security-sensitive

Security Hotspot

Constants should not be used as conditions

Code Smell

"SystemExit" should be re-raised

Code Smell

Bare "raise" statements should only be used in "except" blocks

### String formatting should not lead to runtime errors

Analyze your code

Bug **(1)** Blocker **(?)**

Formatting strings, either with the % operator or str.format method, requires a valid string and arguments matching this string's replacement fields.

This rule raises an issue when formatting a string will raise an exception because the input string or arguments are invalid. Rule {rule:python:S3457} covers cases where no exception is raised and the resulting string is simply not formatted properly.

#### Noncompliant Code Example

```
print('Error code %d' % '42') # Noncompliant. Replace this

print('User {1} is not allowed to perform this action'.format(

print('User {0} has not been able to access {}'.format('Alice

print('User {a} has not been able to access {b}'.format(a='A
```

#### Compliant Solution

```
print('Error code %d' % 42)

print('User {0} is not allowed to perform this action'.format(

print('User {0} has not been able to access {1}'.format('Alice





print('User {a} has not been able to access {b}'.format(a='A
```

#### See

- [Python documentation - Format String Syntax](#)
- [Python documentation - printf-style String Formatting](#)

Available In:

sonarlint | sonarcloud | sonarqube

Recent except blocks
 Code Smell
Comparison to None should not be constant
 Code Smell
"self" should be the first argument to instance methods
 Code Smell
Function parameters' default values should not be modified or assigned
 Code Smell
Some special methods should return