sonar RULES

Products

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...

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
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

Identical expressions should not be used on both sides of a binary operator
Bug

All code should be reachable
Bug

Loops with at most one iteration should be refactored
Bug

Variables should not be self-assigned
Bug

All "except" blocks should be able to catch exceptions
Bug

Regex lookahead assertions should not be contradictory

Analyze your code

Bug | Critical | regex

Lookahead assertions are a regex feature that makes it possible to look ahead in the input without consuming it. It is often used at the end of regular expressions to make sure that substrings only match when they are followed by a specific pattern.

However, they can also be used in the middle (or at the beginning) of a regex. In that case there is the possibility that what comes after the lookahead does not match the pattern inside the lookahead. This makes the lookahead impossible to match and is a sign that there's a mistake in the regular expression that should be fixed.

Noncompliant Code Example

```
r"(?=a)b" # Noncompliant, the same character can't be e
```

Compliant Solution

```
r"(?<=a)b"
r"a(?=b)"
```

Available In:
sonarlint | sonarcloud | sonarqube

Constructing arguments of system commands from user input is security-sensitive

Security Hotspot

Disabling auto-escaping in template engines is security-sensitive

Security Hotspot

Setting loose POSIX file permissions is security-sensitive

Security Hotspot

Formatting SQL queries is security-sensitive

Security Hotspot