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...

Calls should not be made to non-callable values
Bug

Property getter, setter and deleter methods should have the expected number of parameters
Bug

Special methods should have an expected number of parameters
Bug

Instance and class methods should have at least one positional parameter
Bug

Boolean expressions of exceptions should not be used in "except" statements
Bug

Caught Exceptions must derive from BaseException
Bug

Item operations should be done on objects supporting them
Bug

Raised Exceptions must derive from BaseException
Bug

Operators should be used on compatible types
Bug

Function arguments should be passed only once
Bug

Iterable unpacking, "for-in" loops and "yield from" should use an Iterable object
Bug

A secure password should be used when connecting to a database

Analyze your code

Vulnerability | Blocker | cwe owasp

When relying on the password authentication mode for the database connection, a secure password should be chosen.

This rule raises an issue when an empty password is used.

Noncompliant Code Example

Flask-SQLAlchemy

```
def configure_app(app):
    app.config['SQLALCHEMY_DATABASE_URI'] = "postgresql://us
```

Django

```
# settings.py

DATABASES = {
    'postgresql_db': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'quickdb',
        'USER': 'sonarsource',
        'PASSWORD': '', # Noncompliant
        'HOST': 'localhost',
        'PORT': '5432'
    }
}
```

mysql/mysql-connector-python

```
from mysql.connector import connection

connection.MySQLConnection(host='localhost', user='sonarsour
```

Compliant Solution

Flask-SQLAlchemy

```
def configure_app(app, pwd):
    app.config['SQLALCHEMY_DATABASE_URI'] = f"postgresql://u
```

Django

```
# settings.py
import os

DATABASES = {
    'postgresql_db': {
        'ENGINE': 'django.db.backends.postgresql',
```

Variables, classes and functions should be defined before being used

Bug

Identity operators should not be used with dissimilar types

Bug

Only strings should be listed in "__all__"

Bug

"__init__" should not return a value

Bug

```
        'NAME': 'quickdb',
        'USER': 'sonarsource',
        'PASSWORD': os.getenv('DB_PASSWORD'),      # Complia
        'HOST': 'localhost',
        'PORT': '5432'
    }
}
```

mysql/mysql-connector-python

```
from mysql.connector import connection
import os

db_password = os.getenv('DB_PASSWORD')
connection.MySQLConnection(host='localhost', user='sonarsour
```

See

- OWASP Top 10 2021 Category A7 - Identification and Authentication Failures
- OWASP Top 10 2017 Category A2 - Broken Authentication
- OWASP Top 10 2017 Category A3 - Sensitive Data Exposure
- MITRE, CWE-521 - Weak Password Requirements

Available In:

sonarlint | sonarcloud | sonarqube