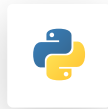


- Secrets
- ABAP
- Apex
- C
- C++
- CloudFormation
- COBOL
- C#
- CSS
- Flex
- Go
- HTML
- Java
- JavaScript
- Kotlin
- Objective C
- PHP
- PL/I
- PL/SQL
- Python**
- RPG
- Ruby
- Scala
- Swift
- Terraform
- Text
- TypeScript
- T-SQL
- VB.NET
- VB6
- XML



## 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...



Unread "private" attributes should be removed

Code Smell

Cognitive Complexity of functions should not be too high

Code Smell

The first argument to class methods should follow the naming convention

Code Smell

Method overrides should not change contracts

Code Smell

Wildcard imports should not be used

Code Smell

String literals should not be duplicated

Code Smell

Functions and methods should not be empty

Code Smell

Server-side requests should not be vulnerable to forging attacks

Vulnerability

Non-empty statements should change control flow or have at least one side-effect

Bug

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

### Server hostnames should be verified during SSL/TLS connections

Analyze your code

Vulnerability Critical cwe privacy owasp ssl

To establish a SSL/TLS connection not vulnerable to man-in-the-middle attacks, it's essential to make sure the server presents the right certificate.

The certificate's hostname-specific data should match the server hostname.

It's not recommended to re-invent the wheel by implementing custom hostname verification.

TLS/SSL libraries provide built-in hostname verification functions that should be used.

#### Noncompliant Code Example

Python [ssl standard](#) library:

```
import ssl

ctx = ssl._create_unverified_context() # Noncompliant: by default
ctx = ssl._create_stdlib_context() # Noncompliant: by default

ctx = ssl.create_default_context()
ctx.check_hostname = False # Noncompliant

ctx = ssl._create_default_https_context()
ctx.check_hostname = False # Noncompliant
```

#### Compliant Solution

Python [ssl standard](#) library:

```
import ssl

ctx = ssl._create_unverified_context()
ctx.check_hostname = True # Compliant

ctx = ssl._create_stdlib_context()
ctx.check_hostname = True # Compliant

ctx = ssl.create_default_context() # Compliant: by default
ctx = ssl._create_default_https_context() # Compliant: by default
```

#### See

- [OWASP Top 10 2021 Category A2](#) - Cryptographic Failures
- [OWASP Top 10 2021 Category A5](#) - Security Misconfiguration
- [OWASP Top 10 2021 Category A7](#) - Identification and Authentication Failures
- [OWASP Top 10 2017 Category A3](#) - Sensitive Data Exposure
- [OWASP Top 10 2017 Category A6](#) - Security Misconfiguration
- [Mobile AppSec Verification Standard](#) - Network Communication Requirements
- [OWASP Mobile Top 10 2016 Category M3](#) - Insecure Communication

 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

- [MITRE, CWE-297](#) - Improper Validation of Certificate with Host Mismatch

Available In:

**sonarlint**  | **sonarcloud**  | **sonarqube** 

© 2008-2022 SonarSource S.A., Switzerland. All content is copyright protected. SONAR, SONARSOURCE, SONARLINT, SONARQUBE and SONARCLOUD are trademarks of SonarSource S.A. All other trademarks and copyrights are the property of their respective owners. All rights are expressly reserved.  
[Privacy Policy](#)