

































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

PL/SQL

## PL/SQL static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your PL/SQL code

All rules

188

 Vulnerability

4

 Bug

45

 Security Hotspot

2

 Code Smell

137

Tags



Search by name...



Cipher algorithms should be robust

 Vulnerability

"COMMIT" should not be used inside a loop

 Bug

Individual "WHERE" clause conditions should not be unconditionally true or false

 Bug

Nullable subqueries should not be used in "NOT IN" conditions

 Bug

"COMMIT" and "ROLLBACK" should not be called from non-autonomous transaction triggers

 Bug

Positional and named arguments should not be mixed in invocations

 Bug

Functions should end with "RETURN" statements

 Bug

Collections should not be iterated in "FOR" loops

 Bug

Using weak hashing algorithms is security-sensitive

 Security Hotspot

Dynamically executing code is security-sensitive

 Security Hotspot

SQL "JOIN" conditions should involve all joined tables

 Code Smell

"SELECT" statements used as argument of "EXISTS" statements should be selective

 Code Smell

"LIKE" clauses should not be used without wildcards

 Code Smell

### Cipher algorithms should be robust

Analyze your code

 Vulnerability

 Critical



 cwe privacy owasp sans-top25

**Strong cipher algorithms** are cryptographic systems resistant to cryptanalysis, they are not vulnerable to well-known attacks like brute force attacks for example.

A general recommendation is to only use cipher algorithms intensively tested and promoted by the cryptographic community.

More specifically for block cipher, it's not recommended to use algorithm with a block size inferior than 128 bits.

#### Noncompliant Code Example

```
PLS_INTEGER := DBMS_CRYPTO.ENCRYPT_DES
               + DBMS_CRYPTO.CHAIN_CBC
               + DBMS_CRYPTO.PAD_PKCS5;
```

#### Compliant Solution

```
PLS_INTEGER := DBMS_CRYPTO.ENCRYPT_AES256
               + DBMS_CRYPTO.CHAIN_CBC
               + DBMS_CRYPTO.PAD_PKCS5;
```

#### See

- [OWASP Top 10 2017 Category A3](#) - Sensitive Data Exposure
- [MITRE, CWE-327](#) - Use of a Broken or Risky Cryptographic Algorithm
- [CERT, MSC61-J](#). - Do not use insecure or weak cryptographic algorithms
- [SANS Top 25](#) - Porous Defenses

Available In:

**sonarlint**



**sonarcloud**



**sonarqube**



Developer  
Edition