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

Go static code analysis

Unique rules to find Bugs, Security Hotspots, and Code Smells in your GO code

All rules 38 | Bug 7 | Security Hotspot 2 | Code Smell 29

Tags

Search by name...

Hard-coded credentials are security-sensitive
Security Hotspot

Cognitive Complexity of functions should not be too high
Code Smell

String literals should not be duplicated
Code Smell

Functions should not be empty
Code Smell

All branches in a conditional structure should not have exactly the same implementation
Bug

"=+" should not be used instead of "+="
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

Variables should not be self-assigned
Bug

Functions should not have identical implementations
Code Smell

Two branches in a conditional structure should not have exactly the same implementation
Code Smell

Related "if/else if" statements should not have the same condition

Analyze your code

Bug | Major | unused pitfall

A chain of `if/else if` statements is evaluated from top to bottom. At most, only one branch will be executed: the first one with a condition that evaluates to `true`.

Therefore, duplicating a condition automatically leads to dead code. Usually, this is due to a copy/paste error. At best, it's simply dead code and at worst, it's a bug that is likely to induce further bugs as the code is maintained, and obviously it could lead to unexpected behavior.

Noncompliant Code Example

```
func example(condition1, condition2 bool) {
  if condition1 {
  } else if condition1 { // Noncompliant
  }
}
```

```
func SwitchWithMultipleConditions(param int) {
  switch param {
  case 1, 2, 3:
    fmt.Println(">1")
  case 3, 4, 5: // Noncompliant; 3 is duplicated
    fmt.Println("<1")
  }
}
```

Compliant Solution

```
func example(condition1, condition2 bool) {
  if condition1 {
  } else if condition2 { // Compliant
  }
}
```

```
func SwitchWithMultipleConditions(param int) {
  switch param {
  case 1, 2, 3:
    fmt.Println(">1")
  case 4, 5: // Compliant
    fmt.Println("<1")
  }
}
```

Available In:

sonarcloud | sonarqube

"switch" statements should not have too many "case" clauses

Code Smell

Track uses of "FIXME" tags

Code Smell

Redundant pairs of parentheses should be removed

Code Smell

Nested blocks of code should not be left empty

Code Smell

Functions should not have too many parameters

Code Smell

Using hardcoded IP addresses is security-sensitive

Security Hotspot

Multi-line comments should not be empty

Code Smell

Boolean checks should not be inverted

Code Smell

Local variable and function parameter names should comply with a naming convention

Code Smell

Boolean literals should not be redundant

Code Smell