




































-  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

Nested blocks of code should not be left empty

Analyze your code

 Code Smell  Major  suspicious

Most of the time a block of code is empty when a piece of code is really missing. So such empty block must be either filled or removed.

### Noncompliant Code Example

```
func compute(a int, b int) {
    sum := a + b
    if sum > 0 { } // Noncompliant; empty on purpose
    fmt.Println("Result:", sum)
}
```

### Compliant Solution











```
func compute(a int, b int) {
    sum := a + b
    if sum > 0 {
        fmt.Println("Positive result")
    }
    fmt.Println("Result:", sum)
}
```

### Exceptions

When a block contains a comment, this block is not considered to be empty. `for` without `init` and `post` statements with empty blocks are ignored as well.

Available In:

sonarcloud  | sonarqube 

<div>"switch" statements should not have too many "case" clauses</div> <div> Code Smell</div>
<div>Track uses of "FIXME" tags</div> <div> Code Smell</div>
<div>Redundant pairs of parentheses should be removed</div> <div> Code Smell</div>
<div>Nested blocks of code should not be left empty</div> <div> Code Smell</div>
<div>Functions should not have too many parameters</div> <div> Code Smell</div>
<div>Using hardcoded IP addresses is security-sensitive</div> <div> Security Hotspot</div>
<div>Multi-line comments should not be empty</div> <div> Code Smell</div>
<div>Boolean checks should not be inverted</div> <div> Code Smell</div>
<div>Local variable and function parameter names should comply with a naming convention</div> <div> Code Smell</div>
<div>Boolean literals should not be redundant</div> <div> Code Smell</div>