rules.sonarsource.com

C++ static code analysis: Immediately dangling references should not be created

1-2 minutes

Binding a temporary object to a reference to const usually lengthens the lifetime of the temporary: instead of being destroyed at the end of the full expression, the temporary will have the same lifetime as the reference itself.

However, lifetime extension is not transitive, so if the definition of the temporary relies on another temporary, this second temporary will always be destroyed at the end of the full expression, creating an immediately dangling reference.

Noncompliant Code Example

```
#include <vector>
```

```
#include <optional>

using namespace std;

void f(int i);

int main() {
    // the vector is a temporary object, and binding
a reference to its first element will not extend the
vector lifetime
    auto const &val = vector{1, 2, 3}[0]; //
Noncompliant, val is an immediately dangling
reference
    f(val);

    int && rval = *std::optional<int>();  //
Noncompliant, rval is an immediately dangling
reference
}
```