C static code analysis: "^" should not be confused with exponentiation

2 minutes

In C and its family of languages, the ^ operator performs the *exclusive or* (xor) operation. This can be misleading, since ^ is also commonly used to designate the exponentiation operation, for instance in BASIC, R or (La)TeX.

This rule will flag uses of ^ in places where an exponentiation is suspected to be the intended operation, i.e on expressions that attempt to *xor* 2 or 10 with a constant expression.

Noncompliant Code Example

```
#include <stdint.h>

uint32_t max_uint16 = 2 ^ 16; // Noncompliant, expression evaluates to 18, instead of the intended 65536
uint32_t one_billion = 10 ^ 9; // Noncompliant, expression evaluates to 3 instead of the intended 1e9
```

Compliant Solution

```
#include <stdint.h>
#include <math.h>

uint32_t max_uint16 = 1 << 16; // Compliant, using left shift to generate a power of 2
uint32_t one_billion = pow(10, 9); // Compliant, using the math pow function
```

Exceptions

No issue will be raised when at least one of the operands is expressed as a binary, octal or hexadecimal literal. In such cases, the assumption is that *xor* operation is intended.

```
#include <stdint.h>

uint32_t using_octal = 02 ^ 016; // Compliant
uint32_t using_binary = 0b10 ^ 9; // Compliant
uint32_t using_hex = 0xFF ^ 0x09; // Compliant
```

See

- Exponentiation on Wikipedia