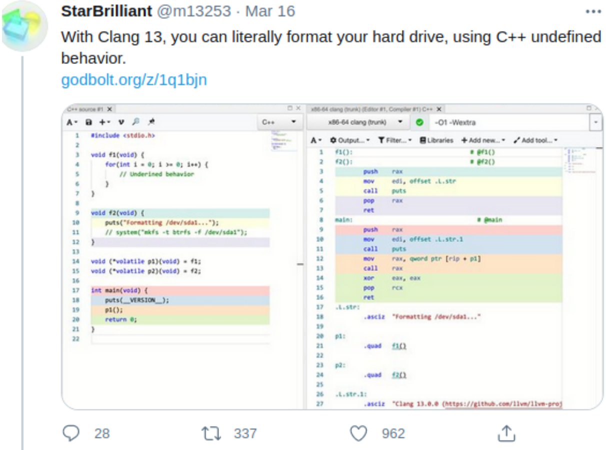
A closer look at default-constructing objects

Adam Graliński

C++ FFFE, May 2021

Today's talk is inspired by this tweet



StarBrilliant @m13253 · Mar 16

With Clang 13, you can literally format your hard drive, using C++ undefined behavior.

godbolt.org/z/1q1bjn

```
#include <stdio.h>

void f1(void) {
    for(int i = 0; i >= 0; i++) {
        // Undefined behavior
    }
}

void f2(void) {
    puts("Formatting /dev/sda1...");
    // system("mkfs -t btrfs -f /dev/sda1");
}

void (*volatile p1)(void) = f1;
void (*volatile p2)(void) = f2;

int main(void) {
    puts(__VERSION__);
    p1();
    return 0;
}
```

-O1 -Wextra

```
f1():                                   # @f1()
f2():                                   # @f2()
        push    rax
        mov     edi, offset .L.str
        call    puts
        pop     rax
        ret
main:                                   # @main
        push    rax
        mov     edi, offset .L.str.1
        call    puts
        mov     rax, qword ptr [rip + p1]
        call    rax
        xor     eax, eax
        pop     rcx
        ret
.L.str:
        .asciz  "Formatting /dev/sda1..."

p1:
        .quad   f1()

p2:
        .quad   f2()

.L.str.1:
        .asciz  "Clang 13.0.0 (https://github.com/llvm/llvm-proj
```

28 337 962

https://twitter.com/m13253/status/1371615680068526081

https://godbolt.org/z/1q1bjn

So what is an undefined behavior, anyway?

undefined behavior - there are *no restrictions* on the behavior of the program. Examples of undefined behavior are data races, memory accesses outside of array bounds, *signed integer overflow*, null pointer dereference, more than one modifications of the same scalar in an expression: without any intermediate sequence point (until C++11) / that are unsequenced (since C++11), access to an object through a pointer of a different type, etc. Compilers are not required to diagnose undefined behavior (although many simple situations are diagnosed), and the compiled program is *not required to do anything meaningful*.

Source: `https://en.cppreference.com/w/cpp/language/ub`, emphasis mine.

Coding time

https://godbolt.org/z/jKh9GaMna

Undefined behavior, reprised

undefined behavior - there are *no restrictions* on the behavior of the program.

- the compiler *really* is allowed to emit *any assembly*...
  - ...one that just crashes or ceases to function
  - ...one that looks good, but has JMPs without corresponding RETs
  - ...one that corrupts memory of another program, but only on friday evenings
- ...and *you, the programmer* are to blame.

Key takeaways

- be aware of the common root causes of UB
- use `volatile` judiciously
- use sanitizers (UBSan) and other fuzzing tools (e.g. Valgrind+memcheck) to detect the triggering of UB
- prefer modern C++ syntax over old C–style syntax — lambdas over naked pointers to functions.

Thank you!