Lesson 2: Debugging

Christian Schwarz, Jakob Krebs

November 24, 2019

Roadmap

basics

gdb

valgrind

basics

types of errors

- compiletime errors
- runtime errors

gdb

gdb

```
# compile our program with debugging symbols
$ gcc -g foo.c -o foo
# open it with gdb
$ gdb foo
```

Breakpoints

```
# set breakpoint at line 12
(gdb) break 12
(gdb) run
[...]
Breakpoint 1, main () at foo.c:12
12 int[]* bla= malloc(20 * sizeof(int));
```

Inspection

```
# how did we end up here?
(gdb) backtrace
(gdb) run
[...]
Breakpoint 1, main () at foo.c:12
12 int[]* bla= malloc(20 * sizeof(int));
```

useful gdb commands

file	load program
r[un]	execute program
b[reak]	set breakpoint
p[rint]	print variable
w[atch]	break and print variable when it changes
n[ext]	execute next line and break
s[tep]	execute next instruction and break
c[ontinue]	execute until next breakpoint
backtrace / **bt**	How did I end up here?

[1]

[1] this table was stolen from
`https://github.com/fsr/c-lessons/blob/master/latex/slides/11_debugging.tex`

valgrind

bugs reported by valgrind

- usage of uninitialized memory
- use after free
- using memory beyond alloced memory
- memory leaks

usage

```
$ valgrind --tool=memcheck --leak-check=yes your_binary
```