COMP 2400 UNIX Tools

Christian Grothoff
christian@grothoff.org

http://grothoff.org/christian/

Motivation

- Debugging is hard
- Some bugs are hard to fix
- Others are hard to find

$\Rightarrow$ Programming language research!

Automatic Program Analysis

- Many program analysis problems are **undecidable**

$\Rightarrow$ Analyzers either underreport, overreport or both

- Program analysis is not a silver bullet

$\Rightarrow$ Use as indicated by your doctor

Types of Program Analyses

- Static analysis:
  - runs at compile time
  - can analyze libraries
  - analyzes entire code
- Dynamic analysis:
  - runs at run-time
  - only analyzes executed code
  - has more information

Valgrind

- Dynamic analysis
- 50x slower execution
- Can find bugs and performance problems
- Free (GPL)!

$\Rightarrow$ Focus for now is on bugs

Using Valgrind

- $ valgrind –tool=TOOLNAME myapplication
- $ valgrind –tool=memcheck myapplication
- $ valgrind –tool=memcheck –leak-check=yes myapplication
- $ valgrind –help

Coverity Prevent

- Static analysis
- Only finds bugs
- Commercial – but free (as in free beer) for you!
- Only available in lab, do not copy!
- Requires `Makefile`

Using Coverity Prevent

- $ export PATH=/opt/prevent-linux-3.0.2/bin/:$PATH
- $ coverity-setup.sh projectname
- $ cd my-project-directory; ./configure
- $ make clean
- $ coverity.sh projectname

Gimpel FlexeLint

- Static analysis
- Source based (considers formatting!)
- Commercial – but free (as in free beer) for you!
- Only available in lab, do not copy!
- Requires some integration work!

Using FlexeLint

- $ export PATH=/usr/local/bin:$PATH
- $ lint.sh mycode.c
- $ xpdf /usr/local/share/doc/flint/flex.pdf

Task

Please write a 200 word summary comparing the three different program analysis tools. Summarize what you liked or disliked about them, how many bugs you found (or did not find) and what you think might be improved.

Questions

?