L1: Make - GDB - Valgrind

CS 283 Systems Programming

Grading: 100 Points

Description

This Lab has three parts, each one designed to help you become familiar with three very important tools: make, GDB, and Valgrind.

Part 1: Make

1. Access the make manual or download the pdf file and read it, starting from this page
   a. https://www.gnu.org/software/make/manual/
2. Download and unzip file make.zip
   a. This file contains several very simple .c and .h files
   b. It also contains several makefiles named makefile-1.txt, makefile-2.txt and so on.
   c. Use the command "make -f makefile-1.txt all" using each makefile separately to complete the make process.
   d. Add comments to each makefile to explain exactly what each line does.
   e. Identify any mistakes in the makefiles or the source (.c, .h) files.

Part 2: GDB

1. Access the GDB manual or download the pdf file and read it, starting from this page
   a. https://www.gnu.org/software/gdb/documentation/
2. Download and unzip file gdb.zip
   a. This file contains three .c programs. Each program has an error that makes it execute incorrectly:
      i. etox.c should approximate e^x but produces an incorrect result.
      ii. g1.c should read characters from the terminal end echo them but fails.
      iii. g2.c should read a line of text from the terminal but causes a segmentation violation.
   b. First, simply compile and run the programs to observe the incorrect behavior.
   c. Second, recompile them to support debugging with gdb and use the gdb commands to identify the source of the incorrect behavior.
   d. Third, fix the errors and run the programs to observe the correct behavior.
   e. Since these programs are very simple, you may be able to identify the errors just by looking at the .c source code. Still, go through the process of debugging to verify that you are familiar with gdb.

f. Note that since g2.c causes a segmentation fault and leaves a core dump file, you can use gdb to look at the program state when it crashed. Use the core command to load a core file. If the core dump file is not generated on segmentation fault it is because by default the limit of core file size is set to 0 blocks. The limits.conf file specifies core dump details. Run the command "ulimit -c unlimited" to allow the system to save the core file.

g. Record all the gdb activity and include it in your lab report. For example, if you are using PuTTY on a Windows machine, then when you select text on the terminal window, PuTTY automatically copies the selected text in the clipboard. Use this feature to copy and paste in your lab report your interactions with gdb.

Part 3: Valgrind

1. Download the Valgrind manual and read it, starting from this page
   a. http://valgrind.org/docs/download_docs.html
2. Download and unzip file valgrind.zip
   a. This file contains three .c programs. Again, each program has an error that may cause incorrect run time behavior.
   b. First, compile and run each program using valgrind.
   c. Second, study the output of valgrind to understand what error is being detected.
   d. Third, fix the error and verify that the programs are running correctly.
   e. Include the output of valgrind and the corrected .c programs in your report.

What To Hand In

1. Copies of the makefiles with your comments as described above.
2. Your interactions with gdb and the corrected .c programs
3. The output of valgrind and the corrected .c programs

Each part is worth 33 points.