CS29206 Systems Programming Laboratory
Spring 2024
Lab Assignment: 3
Date: 08–Feb–2024

Using the GNU debugger gdb

You are given a C source file *guessit.c* and the executable binary file *guessit* generated from this. The executable file is generated with debugging enabled, that is, *guessit.c* is compiled with the -g flag. The code uses some library not accessible to you. Solve the following puzzle using only *guessit* and *guessit.c*, and gdb in the interactive mode.

You should run *guessit* with a single command-line argument: your roll number. This should be a 9-letter string with upper-case letters in the department area: like 21CS10099 (but unlike 21cs10099 or 213CS0099). The code simply exits if you supply no roll number or an invalid roll number.

The code (and the library) uses a secret 20-bit integer s. There is a function *hdistance*(R, g) that takes your roll number R and a guess g for s as arguments, and returns the number of bit positions, at which g differs from s. This return value is called the Hamming distance of two bit strings, justifying the name of the function. Your task is to find out the secret s. You are required to solve this problem by running *guessit* under gdb in the interactive mode.

Follow the instructions given below.

- Download the file *guessit.zip* from Moodle or from the course web site, and unzip it. This will create a subdirectory A3/ with two files *guessit.c* (the C source code) and *guessit* (the binary executable made from the C source). Add execute permission to *guessit*.
- Run *guessit* under gdb.
- You cannot (re)compile *guessit.c* because doing that requires library functions not available to you, so changing *guessit.c* is not an option to you.
- The files except *guessit.c* are not compiled with the -g flag, so do not step into external functions.
- The binary file *guessit* should run on the lab PCs, so you should carry out your experiments there. The binary file may or may not work on other machines or may give different values of the secret s elsewhere. So it is your duty to do the experiment on your lab PC.
- Finding s by any means other than gdb in the interactive mode will not be accepted. You may write a gdb script (if you know what it is and how to do it), but making an exhaustive search over all 20-bit integers will deserve no credit (such a search will take at least an hour anyway).

In the submission server, write the following things in the text box provided. No file submission (source code or screen dump) will be accepted.

- At the beginning, clearly write:

  ```
  My roll number = ...          [This should match your roll number exactly]
  Secret found = ...
  ```

  Note that the secret depends upon your roll number. *Solving the assignment on any other roll number will deserve no credit.*

- This is to be followed by a clear mention of how you used gdb to find the secret s. Specify the exact sequence of gdb commands and scripts (if any) that you use, and when. Note that the sequence may depend on your roll number, and may vary from one student to another *No credits if you do not write this* (*even when the secret is correctly found*).

That's all.