

Tutorial - How to use GDB Debugger for visualizing assembly code?

TA - Aalisha Dalal

April 2019

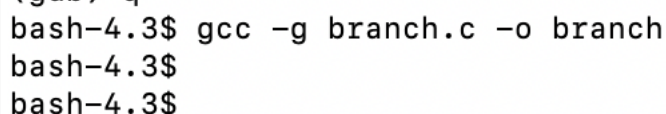
1 Introduction

GDB is a powerful debugging tool that is pre-installed on the Linux SEAS servers. It can be used to analyze program behaviour examine what had caused program failures. It also allows to change values of the variable while running the code in debug mode. The purpose of this short tutorial is to give a short overview of using various commands on the GDB debugger for analysing the assembly code.

2 Method of using the debugger

2.1 Step 1: Executing GCC Command

In order to use the gdb debugger it is essential that we compiled the code using 'gcc' with the -g and -o flag along with the command.



```
bash-4.3$ gcc -g branch.c -o branch
bash-4.3$
bash-4.3$
```

Figure 1: Compiling code with the gcc command

2.2 Step 2: Opening GDB Debugger

The gdb debugger should be opened along with the executable file that you wish to debug/disassemble. For ensuring this, the executable filename is given as an input argument to the gdb command.

```
bash-4.3$  
bash-4.3$ gdb branch  
GNU gdb (GDB) 7.9  
Copyright (C) 2015 Free Software Foundation, Inc.  
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>  
This is free software: you are free to change and redistribute it.  
There is NO WARRANTY, to the extent permitted by law. Type "show copying"  
and "show warranty" for details.  
This GDB was configured as "x86_64-unknown-linux-gnu".  
Type "show configuration" for configuration details.  
For bug reporting instructions, please see:  
<http://www.gnu.org/software/gdb/bugs/>.  
Find the GDB manual and other documentation resources online at:  
<http://www.gnu.org/software/gdb/documentation/>.  
For help, type "help".  
Type "apropos word" to search for commands related to "word"...  
Reading symbols from branch...done.
```

Figure 2: Opening the gdb debugger with the executable filename

2.3 Listing the Code-snippet

The list command can be used to see the entire code within the file.

```
[(gdb) list
1      //
2      //  branch.c
3      //
4      //
5      //  Created by Aalisha on 12/04/19.
6      //
7
8      #include <stdio.h>
9
10     int main()
[(gdb) list
11     {
12         int left = 0;
13         int right = 10;
14         int val = 5;
15
16         int mid = (left + right)/2;
17
18         if(mid == val)
19             return 0;
20         else if(mid < val)
[(gdb) list
21             return -1;
22         else
23             return 1;
24
25     }
[(gdb) list
Line number 26 out of range; branch.c has 25 lines.
```

Figure 3: Using the List command to visualize the code

2.4 Assembly Dump for Function Stack

The code can have various function-routine responsible for different tasks within the code. To visualize the assembly code of a specific function stack, we can use the disassemble command followed by the name of the function.

```
(gdb)
(gdb) list
1      //
2      //  branch.c
3      //
4      //
5      //  Created by Aalisha on 12/04/19.
6      //
7
8      #include <stdio.h>
9
10     int main()
(gdb) list
11     {
12         int left = 0;
13         int right = 10;
14         int val = 5;
15
16         int mid = (left + right)/2;
17
18         if(mid == val)
19             return 0;
20         else if(mid < val)
(gdb) list
21             return -1;
22         else
23             return 1;
24
25     }
(gdb) list
.line number 26 out of range; branch.c has 25 lines.
```

Figure 4: Using the List command to visualize the code

3 Do It Yourself

Certain code examples for C files have been added along with the Tutorial in the folder. You can download those files on your Linux server and follow the instructions in this Tutorial to visualize the assembly code.