

lab 2.2

overview

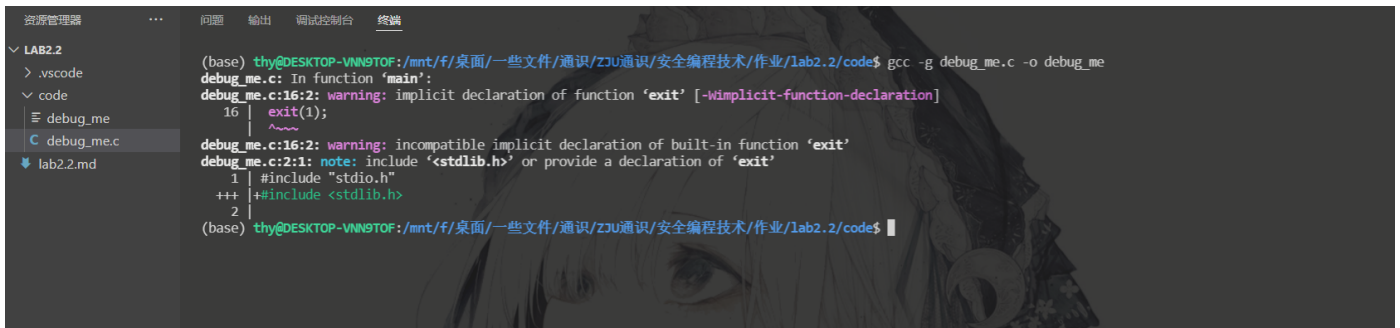
Overview

The lab helps familiarize you with writing a simple Hello World program using C, the GCC compiler link, and Pico(a text editor, link). It uses Ubuntu VM created in Lab 2.1.Here is lab objective:

- Learn to run a program in gcc.
- Learn to debug a program in gdb.

steps

1. 调用gdb, gcc编译并调试 `gcc -g debug_me.c -o debug_me`



```
(base) thy@DESKTOP-VNN9T0F:/mnt/f/桌面/一些文件/通识/ZJU通识/安全编程技术/作业/lab2.2/code$ gcc -g debug_me.c -o debug_me
debug_me.c: In function 'main':
debug_me.c:16:2: warning: implicit declaration of function 'exit' [-Wimplicit-function-declaration]
    16 |     exit(1);
        |     ~~~~
debug_me.c:16:2: warning: incompatible implicit declaration of built-in function 'exit'
debug_me.c:2:1: note: include '<stdlib.h>' or provide a declaration of 'exit'
    1 | #include "stdio.h"
    +++ |+#include <stdlib.h>
    2 |
(base) thy@DESKTOP-VNN9T0F:/mnt/f/桌面/一些文件/通识/ZJU通识/安全编程技术/作业/lab2.2/code$
```

2. 在gdb中运行程序

```
gdb debug_me
run "hello, world" "goodbye, world"
```

```
(base) thy@DESKTOP-VNN9TOF:/mnt/f/桌面/一些文件/通识/ZJU通识/安全编程技术/作业/1ab2.2/code$ gcc -g debug_me.c -o debug_me
debug_me.c: In function 'main':
debug_me.c:16:2: warning: implicit declaration of function 'exit' [-Wimplicit-function-declaration]
   16 |     exit(1);
      |     ~~~~
debug_me.c:16:2: warning: incompatible implicit declaration of built-in function 'exit'
debug_me.c:2:1: note: include '<stdlib.h>' or provide a declaration of 'exit'
   1 | #include "stdio.h"
   ++ |+#include <stdlib.h>
   2 |
(base) thy@DESKTOP-VNN9TOF:/mnt/f/桌面/一些文件/通识/ZJU通识/安全编程技术/作业/1ab2.2/code$ gdb debug_me
GNU gdb (Ubuntu 9.2-0ubuntu1~20.04) 9.2
Copyright (C) 2020 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-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 debug_me...
(gdb) run "hello, world" "goodbye, world"
Starting program: /mnt/f/桌面/一些文件/通识/ZJU通识/安全编程技术/作业/1ab2.2/code/debug_me "hello, world" "goodbye, world"
String '1' - 'hello, world'
String '2' - 'goodbye, world'
Total number of strings: 3
[Inferior 1 (process 55) exited normally]
(gdb) █
```

3. 设置断点

```
[Inferior 1 (process 59) exited normally]
(gdb) break main
Breakpoint 1 at 0x8001199: file debug_me.c, line 10.
(gdb) break print_string
Breakpoint 2 at 0x8001169: file debug_me.c, line 5.
(gdb) █
```

4. 运行到第七行后查看frame

```
Breakpoint 2, print_string (num=0, string=0x7ffffffedd96 "") at debug_me.c:5
5   {
(gdb) next
6   printf("String '%d' - '%s'\n", num, string);
(gdb) where
#0 print_string (num=1, string=0x7ffffffee17a "hello, world") at debug_me.c:6
#1 0x00000000000011fd in main (argc=2, argv=0x7ffffffedeb0) at debug_me.c:21
(gdb) frame0
Undefined command: "frame0". Try "help".
(gdb) frame 0
#0 print_string (num=1, string=0x7ffffffee17a "hello, world") at debug_me.c:6
6   printf("String '%d' - '%s'\n", num, string);
(gdb) frame 1
#1 0x00000000000011fd in main (argc=2, argv=0x7ffffffedeb0) at debug_me.c:21
21   print_string(i, argv[0]); /* function call */
(gdb) █
```

5. next与step的区别

1. step是单步执行，遇到子函数就进入并且继续单步执行。
2. next是在单步执行时，在函数内遇到子函数时不会进入子函数内单步执行，而是将子函数整个执行完再停止，也就是把子函数整个作为一步。