GDB - DEBUGGING EXAMPLE2

Let us write another program that will cause a core dump due to non-initialized memory.

```
#include <iostream>
using namespace std;

void setint(int*, int);
int main()
{
   int a;
   setint(&a, 10);
   cout << a << endl;

   int* b;
   setint(b, 10);
   cout << *b << endl;

   return 0;
}

void setint(int* ip, int i)
{
   *ip = i;
}
```

To enable debugging, the program must be compiled with the -g option.

```
$g++ -g crash.cc -o crash
```

NOTE: We are using g++ compiler because we have used C++ source code.

When you run this program on your linux machine, it will produce the following result:

```
segmentation fault (core dumped)
```

Now let us debug it using gdb:

```
$ gdb crash
(gdb) r
Starting program: /home/tmp/crash
10
10
Program received signal SIGSEGV, Segmentation fault.
0x4000b4d9 in _dl_fini () from /lib/ld-linux.so.2

(gdb) where
#0  0x4000b4d9 in _dl_fini () from /lib/ld-linux.so.2
#1  0x40132a12 in exit () from /lib/libc.so.6
#2  0x4011cdc6 in __libc_start_main () from /lib/libc.so.6

#3  0x080485f1 in _start ()
(gdb)
```

Unfortunately, the program will not crash in either of the user-defined functions, **main** or **setint,** so there is no useful trace or local variable information. In this case, it may be more useful to single-step through the program.

```
(gdb) b main
# Set a breakpoint at the beginning of the function main

(gdb) r
```

```
# Run the program, but break immediately due to the breakpoint.

(gdb) n
# n = next, runs one line of the program

(gdb) n
(gdb) s
setint(int*, int) (ip=0x400143e0, i=10) at crash2.C:20
# s = step, is like next, but it will step into functions.
# In this case the function stepped into is setint.

(gdb) p ip
$3 = (int *) 0x400143e0

(gdb) p *ip
1073827128
```

The value of *ip is the value of the integer pointed to by ip. In this case, it is an unusual value and is strong evidence that there is a problem. The problem in this case is that the pointer was never properly initialized, so it is pointing to some random area in memory $the address 0x40014e0$. By pure luck, the process of assigning a value to *ip does not crash the program, but it creates some problem that crashes the program when it finishes.