通常情況系，程式發生段錯誤時系統會發送 SIGSEGV 信號給程式，缺省處理是退出函數。我們可以使用 signal(SIGSEGV, &your_function);函數來接管 SIGSEGV 信號的處理，程式在發生段錯誤後，自動調用我們準備好的函數，從而在那個函數裡來獲取 當前函式呼叫棧。

舉例如下：

```
#include <stdio.h>
#include <stdlib.h>
#include <stddef.h>
#include <execinfo.h>
#include <signal.h>

void dump(int signo)
{
    void *buffer[30] = {0};
    size_t size;
    char **strings = NULL;
    size_t i = 0;

    size = backtrace(buffer, 30);
    fprintf(stdout, "Obtained %zd stack frames.nm\n", size);
    strings = backtrace_symbols(buffer, size);
    if (strings == NULL)
    {
        perror("backtrace_symbols.");
        exit(EXIT_FAILURE);
    }

    for (i = 0; i < size; i++)
    {
        fprintf(stdout, "%s\n", strings[i]);
    }
    free(strings);
    strings = NULL;
    exit(0);

}
```

```
void func_c()
{
    *((volatile char *)0x0) = 0x9999;
}

void func_b()
{
    func_c();
}

void func_a()
{
    func_b();
}

int main(int argc, const char *argv[])
{
    if (signal(SIGSEGV, dump) == SIG_ERR)
        perror("can't catch SIGSEGV");
    func_a();
    return 0;
}
```

編譯器：

```
gcc -g -rdynamic test.c -o test; ./test
```

輸出如下：

```
Obtained6stackframes.nm
./backstrace_debug(dump+0x45)[0x80487c9]
[0x468400]
./backstrace_debug(func_b+0x8)[0x804888c]
./backstrace_debug(func_a+0x8)[0x8048896]
./backstrace_debug(main+0x33)[0x80488cb]
/lib/i386-linux-gnu/libc.so.6(__libc_start_main+0xf3)[0x129113]
```

接著：

```
objdump -d test > test.s
```

在 test.s 中搜索 804888c 如下：

```
8048884 <func_b>:
8048884: 55              push %ebp
8048885: 89 e5         mov %esp, %ebp
8048887: e8 eb ff ff ff         call 8048877 <func_c>
804888c: 5d                pop %ebp
804888d: c3                ret
```

其中 80488c 時調用（call 8048877）C 函數後的位址，雖然並沒有直接定位到 C 函數，通過彙編代碼， 基本可以推出是 C 函數出問題了（pop 指令不會導致段錯誤的）。
我們也可以通過 addr2line 來查看

```
addr2line 0x804888c -e backstrace_debug -f
```

輸出：

```
func_b
/home/astrol/c/backstrace_debug.c:57
```