How to Backtrace the Stack within a C Function in Linux

Say that a problem occurs in function A within a C code and we want to know the name of function B which called function A leading to this problem. On Linux, this can be found using backtrace as follows. Insert the following code before function A:

```
#include <execinfo.h>

void print_trace(void) {
  void* callstack[128];
  int i, frames = backtrace(callstack, 128);
  char** strs = backtrace_symbols(callstack, frames);
  for (i = 0; i < frames; ++i) {
    printf("%s\n", strs[i]);
  }
  free(strs);
}
```

Also, make sure to compile with debugging symbols using the -g flag and to add the command line option -rdynamic to gcc at the linking stage. Thus, if compiling CFDWARP, you should add the -g flag to CFLAGS and to LFLAGS in CFDWARP/.makefileheader. You should also add -rdynamic within the LFLAGS. Call the function print_trace() just before the problem occurs in function A. Recompile, relink, and execute the code to find the backtrace.