Subject: Call_External and referenced symbol not found
Posted by Todd David halter[1] on Tue, 07 May 2002 18:00:34 GMT
View Forum Message <> Reply to Message

I am working on a Sun SPARC Ultra-2 with OS 5.7 and IDL 5.4.
I am using call_external to call an external C function. I have made my so file and can call my test functions within the so file just fine. The problem comes when I try to call a function from my so that is located in a .a file.

example:

```
37 long
38 zinit (int argc, void *argv[])
39 {
40   IDL_STRING processName;
41
42   FILE *fp;
43
44   processName = *(IDL_STRING *)argv[0];
45
46   fp = fopen ("zlog.debug", "a");
47   fprintf (fp, "processName = %s\n", processName.s);
48   fclose (fp);
49
50   /* Set up connections to zeb */
51   usy_init (); /* located in libZeb.a */
52
53   return (TRUE);
54 } /* end zinit (...) */
```

This function works fine until I add line 51. When I add this line I get the
error (in IDL):

```
ld.so.1: /apps/base/rsi/idl_5.4/bin/bin.solaris2.sparc64/idl: fatal:
relocation error: file ../../bin/zlog.so: symbol usy_init: referenced
symbol not found
Killed
```

My IDL term crashes and I have to log out of the system.
Do I need to make libZeb.a into a so file?
Any help would be greatly appreciated.
Thanks.

Todd Halter, todd*DOT*halter*AT*pnl*DOT*gov
Pacific Northwest National Laboratory

Subject: Re: Call_External and referenced symbol not found
Posted by Malcolm Walters on Wed, 08 May 2002 09:01:08 GMT
View Forum Message <> Reply to Message

"Todd David halter" <thalter@beta.tricity.wsu.edu> wrote in message
news:ab94o1$fhu3$1@murrow.it.wsu.edu...
> Do I need to make libZeb.a into a so file?
> Any help would be greatly appreciated.
> Thanks.

I was in the middle of a long response detailing archive vs. shared object behaviour but I think this is straight forward.
When creating your shared object are you linking it against libZeb.a? If you are not then try adding a '-lZeb' to your link step.
Otherwise I will post my longer explanation about shared objects.
Malcolm

Subject: Re: Call_External and referenced symbol not found
Posted by Malcolm Walters on Thu, 09 May 2002 14:41:58 GMT
View Forum Message <> Reply to Message

"Todd David halter" <thalter@beta.tricity.wsu.edu> wrote in message
news:abbhks$21ov$1@murrow.it.wsu.edu...
> I do have the -lZeb and it still does not work. If you would post you
> long re: that would be great.
>
> Todd

Todd, I'm not 100% sure exactly what you are doing, I mainly work on Linux but this should transfer ok.

What I have done before on several occasions is;
1) Start with an archive file. (libA.a)
2) Write functions calling this library (B.c) and compile using something of the form
gcc -shared -o libB.so B.c -lA
3) Call the wrapper functions in libB from within IDL.

Note that all the functions you are going to call in 'libA.a' from IDL must be referenced in 'B.c'. If you miss any out then the resulting shared object does not contain the functions and they cannot be called. This can be done in a dummy routine that you will never call if necessary. All call_externals should call 'libB.so'.

If you have the source for libA then recompile it to give a shared object.
This way B.c only needs to make calls that actually do something.
Then if you wish to call functions in libA then you must call these directly

(call_external,'libA.so',...)
You will need to have both libraries on your 'LD_LIBRARY_PATH' environmental variable.

Note that either of these should give you only one instance of each shared library. If you have an archive do not try something like
gcc -shared -o libOpenB.so OpenB.c -lA
gcc -shared -o libCloseB.so CloseB.c -lA
if you need internal data within libA to be preseved between the open and close calls.

Malcolm

Subject: Re: Call_External and referenced symbol not found
Posted by Nigel Wade on Mon, 13 May 2002 09:03:27 GMT
View Forum Message <> Reply to Message

Todd David halter wrote:

> I made a mistake in my initial post. The function that is giving me
> trouble is in libsocket.a. From my so file (zlog.so) I am calling the
> function usy_init () which is located is libsocket.a, which is located in
> /usr/lib. The way I am compiling it is:
>
> cc -g -G -K pic -xarch=v9 -I/apps/base/rsi/idl_5.4/external -c zlog.c -o
> obj/zlog.o
> cc -B static -G -K pic -xarch=v9 -L/usr/lib -lsocket -o zlog.so
> obj/zlog.o

^^^^^^^^

>
> The v9 tell it to create 64bit code not 32bit. I am still getting the
> same error. I will keep playing with it, but if anyone can see what I am
> doing wrong please let me know. Thanks.
>
> Todd

I think the problem is the order of the arguments to cc (and therefore to ld).

The arguments to ld (which is used by cc to do the linking) are order dependent. Libraries are scanned in the position in which they occur on the command line, and they are scanned only once. At the time libsocket is scanned, the externals in zlog.o are not undefined, so ld does not include them in the resultant DSO. Try changing the order of the arguments to cc, placing libraries *after* the objects.

--

Nigel Wade, System Administrator, Space Plasma Physics Group,
University of Leicester, Leicester, LE1 7RH, UK
E-mail : nmw@ion.le.ac.uk
Phone : +44 (0)116 2523568, Fax : +44 (0)116 2523555