Gromacs - Bug #1535

osx C++ symbol visibility warnings

06/27/2014 07:55 PM - Mark Abraham

<table>
<thead>
<tr>
<th>Status:</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Low</td>
</tr>
<tr>
<td>Assignee:</td>
<td></td>
</tr>
<tr>
<td>Category:</td>
<td></td>
</tr>
<tr>
<td>Target version:</td>
<td>5.0</td>
</tr>
<tr>
<td>Affected version - extra info:</td>
<td>5.0</td>
</tr>
</tbody>
</table>

**Description**

On bs_mac (Darwin Kernel Version 11.4.2), with cmake 2.8.11.2 or 3.0.0, icc 13 and release-5-0 (going back at least as far as 75b834174109b88, and even a merge including the RPATH fixes from release-4-6 such as ae3944ce092), we get a very large number of linker warnings issued about C++ symbol visibility, e.g.

```
ld: warning: direct access in __ZNK5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St11range_error>::7rethrowEv to global weak symbol __ZN5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St11range_error>::11Ev means the weak symbol cannot be overridden at runtime. This was likely caused by different translation units being compiled with different visibility settings.
ld: warning: direct access in __ZNK5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St11logic_error>::7rethrowEv to global weak symbol __ZN5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St11logic_error>::11Ev means the weak symbol cannot be overridden at runtime. This was likely caused by different translation units being compiled with different visibility settings.
ld: warning: direct access in __ZNK5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St12length_error>::7rethrowEv to global weak symbol __ZN5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St12length_error>::11Ev means the weak symbol cannot be overridden at runtime. This was likely caused by different translation units being compiled with different visibility settings.
ld: warning: direct access in __ZN5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St12domain_error>::7rethrowEv to global weak symbol __ZN5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St12domain_error>::11Ev means the weak symbol cannot be overridden at runtime. This was likely caused by different translation units being compiled with different visibility settings.
```

Id: warning: direct access in

```
__ZNK5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St11range_error>::7rethrowEv
```

to global weak symbol

```
__ZN5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St11range_error>::11Ev
```

means the weak symbol cannot be overridden at runtime. This was likely caused by different translation units being compiled with different visibility settings.

Id: warning: direct access in

```
__ZNK5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St11logic_error>::7rethrowEv
```

to global weak symbol

```
__ZN5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St11logic_error>::11Ev
```

means the weak symbol cannot be overridden at runtime. This was likely caused by different translation units being compiled with different visibility settings.

Id: warning: direct access in

```
__ZNK5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St12length_error>::7rethrowEv
```

to global weak symbol

```
__ZN5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St12length_error>::11Ev
```

means the weak symbol cannot be overridden at runtime. This was likely caused by different translation units being compiled with different visibility settings.

Id: warning: direct access in

```
__ZNK5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St12domain_error>::7rethrowEv
```

to global weak symbol

```
__ZN5boost16exception_detail10clone_implINS0_39current_exception_std_exception_wrapper<St12domain_error>::11Ev
```

means the weak symbol cannot be overridden at runtime. This was likely caused by different translation units being compiled with different visibility settings.
The underlying symbols come from at least thread-MPI, boost, gmock. Issues discussed at https://gcc.gnu.org/wiki/Visibility are probably relevant.

Unilateral use of -fvisibility=extern per

Using cmake 2.8.11.2 did not help so probably
http://www.kitware.com/blog/home/post/510 and
http://www.cmake.org/Wiki/CMake_RPATH_handling are not the relevant issue.

My cmake commands were like
CC=icc CXX=icpc /opt/cmake/2.8.11.2/bin/cmake -DCMAKE_BUILD_TYPE=Debug
-DCMAKE_INSTALL_PREFIX=install -DGMX_THREAD_MPI=off -G Ninja ..

mdrun still seems to work, but I am not sure what the problem is, so hard to assess severity.

Related issues:
Duplicates Gromacs - Task #1483: Fix warnings ignored by Jenkins Closed 04/17/2014

History
#1 - 06/27/2014 07:56 PM - Mark Abraham
- Description updated

#2 - 06/27/2014 08:02 PM - Mark Abraham
Turning thread-MPI on didn't help either

#3 - 06/27/2014 08:37 PM - Teemu Murtola
- Duplicates Task #1483: Fix warnings ignored by Jenkins added

#4 - 06/29/2014 05:43 PM - Teemu Murtola
I took a brief look at this, but without access to icc, it is difficult to find out the actual cause (can be difficult even with access...).

More or less all linker invocations seem to produce some amount of these, including libgromacs and all tests. Since this is a non-gpu build, there shouldn't be anything in our build system that could cause an internal inconsistency, since libgromacs is a single add_library call (and making the test...
libraries into a single link step, removing all static libraries, does not remove any warnings there either).

The mentioned symbols are mainly related to boost and Google Test, but there is also ios_base. Not sure if there is anything directly from our code, except for an exception class from thread-mpi. Some possible paths would be to

- inspect the headers for whether these symbols have something in common wrt. declared visibility, and whether icc actually defines those gcc compatibility defines that at least boost is using to determine what visibility attributes to use,
- compare the objects produced by icc and clang for differences in symbol visibility (using otool), and
- check whether inlining has something to do with this.

These could be easier if one could construct a simpler test case than full Gromacs that would still reproduce the warnings.

The warning is probably harmless during runtime, except possibly if some system library is upgraded without rebuilding Gromacs.

---

**#5 - 07/16/2014 03:53 PM - Roland Schulz**

I suggest we upgrade ICC on Mac. Currently 13.0.1 (2013 Update 1) is installed. The latest 13 version for Mac is 13.0.3 (2013 Update 5). The problem could be that ICC is older than the version of Xcode we have.

---

**#6 - 07/24/2014 06:52 PM - Mark Abraham**

Roland Schulz wrote:

I suggest we upgrade ICC on Mac. Currently 13.0.1 (2013 Update 1) is installed. The latest 13 version for Mac is 13.0.3 (2013 Update 5). The problem could be that ICC is older than the version of Xcode we have.

I poked Erik about licenses again

---

**#7 - 08/11/2014 12:13 AM - Stefan Fleischmann**

Roland Schulz wrote:

I suggest we upgrade ICC on Mac. Currently 13.0.1 (2013 Update 1) is installed. The latest 13 version for Mac is 13.0.3 (2013 Update 5). The problem could be that ICC is older than the version of Xcode we have.

Installed 2013 Update 5. I believe the warnings are gone, see here. CMake reports the compiler version as 13.0.0.20130606 but it is version 13.0.3.198 build 20130606.
#8 - 08/18/2014 02:42 AM - Roland Schulz
- Status changed from New to Closed

#9 - 08/18/2014 02:42 AM - Roland Schulz
I agree that all warnings are gone.