fhicl-cpp - Feature #11860

fhiclcpp-types tests fail for GCC5/Clang due to inline namespaces in demangled symbols

03/02/2016 10:38 AM - Ben Morgan

Status:	Closed	**Start date:**	03/02/2016
Priority:	Normal	**Due date:**	
Assignee:	Kyle Knoepfel	**% Done:**	100%
Category:		**Estimated time:**	1.00 hour
Target version:	2.00.01	**Spent time:**	0.50 hour

Description

Tests in tests/fhiclcpp-types fail on Clang and GCC5 as the strings returned by cetlib::demangle_symbol include the (new for GCC5) inline namespace and full type specification. For example
with GCC5

```
$ ctest -V -I 79,79
...
    Start 79: conditional_t
...
79: Environment variables:
79:  FHICL_FILE_PATH=/<path>/fnal-fhicl-cpp.git/test:.
79: Test timeout computed to be: 9.99988e+06
79: conditional_t.out
79: conditional_t.out-filtered
79:
79: *** No errors detected
79: Comparison of filtered output conditional_t.out-filtered with /<path>fnal-fhicl-cpp.git/test/f
hiclcpp-types/conditional_t-ref.txt failed:
79: --- /<path>/test/fhiclcpp-types/conditional_t-ref.txt    2016-03-02 12:00:07.000000000 +0000
79: +++ conditional_t.out-filtered    2016-03-02 16:21:57.000000000 +0000
79: @@ -16,9 +16,9 @@
79:    -          <int>
79:    -       ]
79:    -
79: - -     ( boxName: <string>
79: + -     ( boxName: <__cxx11::basic_string<char, char_traits<char>, allocator<char> >>
79:    -
79: - -       material: <string>
79: + -       material: <__cxx11::basic_string<char, char_traits<char>, allocator<char> >>
79:    -    }
79:
79:
79:
79: CMake Error at /<path>/SuperArt.gcc5/share/cetbuildtools2/cmake/Modules/RunAndCompare.cmake:58
 (message):
79:   Error comparing conditional_t.out-filtered and
79:   /<path>/fnal-fhicl-cpp.git/test/fhiclcpp-types/conditional_t-ref.txt.
79: Call Stack (most recent call first):
79:   /<path>/SuperArt.gcc5/share/cetbuildtools2/cmake/Modules/RunAndCompare.cmake:100 (filter_and
_compare)
79:
79:
79:
1/1 Test #79: conditional_t ....................***Failed    0.11 sec
```

GCC 4.9 demangles the full basic_string type back to std:string o.k.

I've reported this here as this cetlib::demangle_symbol isn't doing anything wrong that I can see for Clang/GCC/cxxabi (the correct, albeit full, demangled type is returned), and am not sure how tied to demangling fhiclcpp is for functionality (this part appears to be for info/documentation?)

History

#1 - 03/02/2016 11:15 AM - Marc Paterno

- Tracker changed from Bug to Feature

I've changed the 'tracker' from 'bug' to 'feature' (for a feature request), since the essence of the issue is that a compiler we don't yet support yields a test failure.

We've been waiting on moving to GCC 5.x for two reasons (1) no demand from the stakeholders (it is usually us who pushes to move to a new compiler, not the experiments) and (2) ROOT 6 has trouble with the new ABI of GCC 5.x.

The nature of the 'trouble' is that ROOT's dictionary mechanisms expect certain names for types in the C++ Standard Library, and has not yet accommodated to the changes in GCC 5. We believe that instructing GCC 5 to use the GCC 4 ABI solves this problem. However, we don't know whether using the GCC 4 ABI with GCC 5 removes some of the important reasons for moving to GCC 5: the modern C++ support that GCC could only implement by breaking the ABI.

While we are not pushing a move to support GCC 5 at this time, if you want to try building using GCC 5 and the GCC 4 ABI, we'd be happy to take patches that fix any problems you encounter. We'd also be happy to accept any patches to that will allow the Clang build to behave as the tests expect.

Please note in this case we would not want to change the test. This test is verifying that the user is being shown the name of a type he will understand (and not something like __cxx11::basic_string<char, char_traits<char>, allocator<char> >. We would not want to change what the cet::demange_symbol function returns; we would want the description printing system to understand the answers from different compilers, and to translate each into a human-friendly result.

#2 - 03/02/2016 12:02 PM - Ben Morgan

Marc Paterno wrote:

> I've changed the 'tracker' from 'bug' to 'feature' (for a feature request), since the essence of the issue is that a compiler we don't yet support yields a test failure.
>
> We've been waiting on moving to GCC 5.x for two reasons (1) no demand from the stakeholders (it is usually us who pushes to move to a new compiler, not the experiments) and (2) ROOT 6 has trouble with the new ABI of GCC 5.x.
>
> The nature of the 'trouble' is that ROOT's dictionary mechanisms expect certain names for types in the C++ Standard Library, and has not yet accommodated to the changes in GCC 5. We believe that instructing GCC 5 to use the GCC 4 ABI solves this problem. However, we don't know whether using the GCC 4 ABI with GCC 5 removes some of the important reasons for moving to GCC 5: the modern C++ support that GCC could only implement by breaking the ABI.
>
> While we are not pushing a move to support GCC 5 at this time, if you want to try building using GCC 5 and the GCC 4 ABI, we'd be happy to take patches that fix any problems you encounter. We'd also be happy to accept any patches to that will allow the Clang build to behave as the tests expect.
>
> Please note in this case we would not want to change the test. This test is verifying that the user is being shown the name of a type he will understand (and not something like __cxx11::basic_string<char, char_traits<char>, allocator<char> >. We would not want to change what the cet::demange_symbol function returns; we would want the description printing system to understand the answers from different compilers, and to translate each into a human-friendly result.

Thanks Marc! On the last point, I'm happy to take a look at this for Clang at least, though could you clarify the tasks of the demangling/description printing functions? I read the above as

1. cet::demangle_symbol must always return the direct demangled name without modification
    1. It would be compiler/cxxabi dependent, so one could get "std::string" on GCC 4.9 but "std::__cxx11::basic_string<char, char_traits<char>, allocator<char> >" for GCC 5 (cxx11 ABI)
2. Clients of cet::demangle_symbol are responsible for any translation of its return value for further use

Any requirements/specifications for the human-friendly part when dealing with typedefs/default template parameters (e.g. even in GCC 4.9, demangling std::vector<double> prints "std::vector<double, std::allocator<double> >")?

#3 - 03/07/2016 11:33 AM - Kyle Knoepfel

Points 1 and 2, as you list them, are correct. For the particular test suite for which you are experiencing failures, the only printed type that presents any difficulties is std::string--we print the types for only atomic parameters: numeric-types (int, float, double, etc.), bool, and string. So (e.g.) std::vector shouldn't be an issue for the fhiclcpp tests.

#4 - 03/07/2016 11:33 AM - Kyle Knoepfel

- Status changed from New to Feedback

#5 - 04/01/2016 05:42 AM - Ben Morgan

- File 0001-Preliminary-fix-for-Issue-11860.patch added

Attached is a preliminary patch to fhiclcpp's expected_types struct that resolves the inline namespace issue via a template specialisation. The specialised struct uses the post-processing functions as per the non-specialised struct, though that may not be the optimum implementation.

With this in place, all tests pass on both clang (Apple, libc++) and gcc 5 (Homebrew/Mac/Linux, libstdc++).

#6 - 04/04/2016 11:35 AM - Kyle Knoepfel

Thanks, Ben. We'll test and apply.

#7 - 04/04/2016 11:36 AM - Kyle Knoepfel

- Status changed from Feedback to Assigned

- Assignee set to Kyle Knoepfel

- Estimated time set to 1.00 h

#8 - 05/16/2016 10:01 AM - Kyle Knoepfel

- Status changed from Assigned to Resolved

- Target version set to 2.00.01

Patch applied cleanly. Implemented with fhicl-cpp:119e71b9.

Thanks, Ben.

#9 - 05/16/2016 10:01 AM - Kyle Knoepfel

- % Done changed from 0 to 100

#10 - 05/19/2016 03:22 PM - Kyle Knoepfel

- Status changed from Resolved to Closed

Files

0001-Preliminary-fix-for-Issue-11860.patch	1.59 KB	04/01/2016	Ben Morgan