



PathScale™ Compiler Suite Support Guide

Version 3.2

Information furnished in this manual is believed to be accurate and reliable. However, PathScale LLC assumes no responsibility for its use, nor for any infringements of patents or other rights of third parties which may result from its use. PathScale LLC reserves the right to change product specifications at any time without notice. Applications described in this document for any of these products are for illustrative purposes only. PathScale LLC makes no representation nor warranty that such applications are suitable for the specified use without further testing or modification. PathScale LLC assumes no responsibility for any errors that may appear in this document.

No part of this document may be copied nor reproduced by any means, nor translated nor transmitted to any magnetic medium without the express written consent of PathScale LLC. In accordance with the terms of their valid PathScale agreements, customers are permitted to make electronic and paper copies of this document for their own exclusive use.

Linux is a registered trademark of Linus Torvalds.

PathScale, the PathScale logo, and EKOPath are registered trademarks of PathScale, LLC.

Red Hat and all Red Hat-based trademarks are trademarks or registered trademarks of Red Hat, Inc.

SuSE is a registered trademark of SuSE Linux AG.

All other brand and product names are trademarks or registered trademarks of their respective owners.

© 2007 PathScale, LLC. All rights reserved.
© 2006, 2007 QLogic Corporation. All rights reserved worldwide.
© PathScale 2004, 2005, 2006. All rights reserved.
First Published: April 2004
Printed in U.S.A.

PathScale LLC, 2071 Stierlin Ct., Suite 200, Mountain View, CA 94043.

Table of Contents

Section 1

Introduction

1.1	Overview	1-1
1.2	Conventions Used in this Document.	1-1
1.3	Documentation Suite	1-1

Section 2

Getting Support

2.1	Support Online	2-1
2.2	Email Support	2-1

Section 3

Submitting a Bug Report

3.1	Filing a bug	3-1
3.1.1	The <code>pathbug-helper</code> Tool	3-1
3.1.2	The <code>pathhow-compiled</code> Tool	3-2
3.1.3	Other Information	3-2
3.2	Compiler Crashes	3-3
3.2.1	Reporting Compiler Crashes: C and C++	3-4
3.2.2	Reporting Compiler Crashes: Fortran	3-4
3.3	Compiler Error, but the Code is Correct	3-4
3.4	Code Compiles and Links, But Gets the Wrong Answer	3-5
3.5	Other Code Faster than PathScale Compiler's Code	3-5

3.6 Working Around Bugs 3-5

3.7 Release Dates and Bug Fixes 3-6

3.8 Proprietary or Confidential Source Code 3-6

Notes

Section 1 Introduction

1.1 Overview

This document describes how to get support for the PathScale™ Compiler Suite. The introduction provides an overview of this guide and the PathScale Compiler Suite documentation. The Support Guide is organized into these sections:

- [Section 2](#) describes options for support from PathScale
- [Section 3](#) provides guidelines for filing a bug

1.2 Conventions Used in this Document

These conventions are used throughout this document.

Convention	Meaning
<code>command</code>	Fixed-space font is used for literal items such as commands, files, routines, and pathnames.
<i>variable</i>	Italic typeface is used for variable names or concepts being defined
user input	Bold, fixed-space font is used for literal items the user types in. Output is shown in non-bold, fixed-space font.
\$	Indicates a command line prompt
#	Indicate a root command line prompt
[]	Brackets enclose optional portions of a command or directive line.
...	Ellipses indicate that a preceding element can be repeated.
NOTE:	Indicates important information

1.3 Documentation Suite

The PathScale Compiler Suite product documentation set includes:

- *The PathScale Compiler Suite and Subscription Manager Install Guide*
- *The PathScale Compiler Suite User Guide*
- *The PathScale Debugger User Guide*
- *The PathScale Compiler Suite Support Guide*

There are also online manual pages (“man pages”) available describing the flags and options for the PathScale Compiler Suite. These man pages are shipped with

-

the Compiler Suite: `eko`, `pathf95`, `pathcc`, `pathCC`. The `pathscale-intro` man page gives an overview of all the various man pages that are included with the Compiler Suite.

Please see the PathScale web sites for further information about current releases and developer support.

<http://www.pathscale.com>

Section 2

Getting Support

PathScale offers several options for support for the PathScale Compiler Suite.

2.1

Support Online

Support web pages are found on the PathScale web site.

<http://www.pathscale.com/support.html>

The general PathScale pages are supplemented by a private support website. You may access the private support website using your email address and the password sent when you registered for a trial license. If you do not have a password, please contact `support@pathscale.com` or re-register for a trial license.

The private support website contains links to the documents for recent versions of the compiler code. Following these links will take you to the available documentation.

2.2

Email Support

You can email `support@pathscale.com` at any time. Someone from our customer support department will be in contact with you within one business day. Usually, someone will respond during the same day, allowing for time zone differences.

We encourage you to check the support web sites frequently. We update them often with information, tips, and news about releases.

-



Notes

Section 3

Submitting a Bug Report

Our goal at PathScale is to produce fast, bug-free compilers. To that end, we have an extensive in-house test suite, and we encourage our customers to submit bug reports against our compiler. We appreciate our customers taking the trouble to submit bugs, and look forward to working with you!

3.1 Filing a bug

We divide bugs into the following categories:

1. The compiler crashes while compiling your code. These are often the easiest to replicate and fix, but hard to work around.
2. The compiler gives an error compiling your code, but you think your code is correct.
3. Your code compiles and links, but it gets the wrong answer.
4. Your code compiles and links, but will not run.
5. Performance bugs. Another compiler produces significantly faster code than PathScale's compiler.

In order to fix the bug, we need to get enough information from you to understand the context of the bug. The best way to provide this information is to use the `pathbug-helper` tool.

3.1.1 The `pathbug-helper` Tool

The best way to gather a more exhaustive set of information we might need about your situation is to use our `pathbug-helper` tool. We designed `pathbug-helper` to collect information about your compiler environment to aid our support team in diagnosing and solving your compiler problems.

The `pathbug-helper` tool is installed along with the rest of the compiler RPMs, into the `<install_directory>` (`/opt/pathscale/bin` by default) on your machine.

To run `pathbug-helper`, using the same machine you were compiling on (from the `<install_directory>`), type:

```
$ pathbug-helper > info-for-pathsacle.txt
```

This will generate an output file called `info-for-pathsacle.txt`. Include this output file in your bug report to `support@pathsacle.com`.

3.1.2

The pathhow-compiled Tool

The `pathhow-compiled` tool can be used to list the compiler options that are used to compile the program. The tool lists the options given by the user as well as important options are default for your system.

The `pathhow-compiled` tool is installed along with the rest of the compiler RPMs, into the `<install_directory>` (`/opt/pathscale/bin` by default) on your machine.

To run `pathhow-compiled` on an executable `test` generated by the PathScale compiler, type:

```
$ pathhow-compiled test
```

This will produce an output such as:

```
PathScale Compiler Version 3.1 compiled test.c with options:
-O2 -march=athlon64 -msse2 -mno-sse3 -mno-3dnow -mno-sse4a -m64
```

The `pathhow-compiled` tool also works on object (`.o`) files generated by the PathScale compiler.

3.1.3

Other Information

If you choose not to use `pathbug-helper`, we will need you to collect information about your system and your application. This information should be included in all bug reports:

- Which Linux release? You can find this by:

```
$ cat /etc/issue
```
- Which Linux kernel? You can find this by:

```
$ cat /proc/version
```
- Which PathScale compiler release? If you installed from RPMs:

```
$ rpm -q -a | grep _psc
```

Alternately you can use this command:

```
$ pathcc -v
```

or

```
$ pathcc -version
```

We probably won't need to know about your hardware configuration, but if we do, we'll ask.

Other information may be needed, depending upon the circumstances. Please review the categories below for more details.

3.2

Compiler Crashes

The compiler should not crash. If it does, it may be caused by:

- Compiler code error. This is something we need to fix.
- Input (source) or compiler flags and settings that cause the compiler to loop forever (a kind of crash). This is something we need the compiler to accommodate; we will work with you to find a solution.
- System resource limits (such as virtual memory) or filesystem errors. This is something you may need to adjust on your system.

If the compiler encounters an internal error while it is compiling a source file, it will generate and save a *problem report file*, and print the name of the problem report file it has saved.

This problem report file contains a description of the error and how the compiler was invoked. It also contains a preprocessed copy of the source code that may have caused the internal error. If possible, please attach the problem report file when reporting a problem to PathScale support.

Here's an example:

```
$ pathCC -O2 -msse2 -ftemplate-depth-60 my_crashing_code.cpp
Internal compiler error: Segmentation fault
Please submit a full bug report, with preprocessed source if
appropriate.
See <URL:http://www.pathscale.com/support.html> for instructions.
pathCC INTERNAL ERROR: /opt/pathscale/lib/3.0/gficc returned
non-zero status 1
Preprocessed source saved into
/home/.ekopath-bugs/pathCC_bug_kZbERe. ii
Please attach this file to your bug report, if possible.
```

The preprocessed source file has a random name, and includes a few lines of detailed information at the beginning (the first line is the compiler command line, with file names stripped out):

```
// pathCC -O2 -msse2 -ftemplate-depth-60 -default_options
// PathScale EKOPath(TM) Version 3.0
// ChangeSet bos@camp4.serpentine.com|ChangeSet|20050402064013|
// 07361
// Built by bos@camp4.serpentine.com in /home/pathscale
// Build date 2007-01-12 01:40:59 -0800
```

The variable called `PSC_CPP_SAVE_DIR` gives the name of the directory in which to save the file. You can set this variable to a unique directory, then check in there after a "make -k" has completed to see if there were any compiler errors.

Please, always report compiler crashes, even if you can work around them by reducing optimization.

In order to replicate your bug, we will need the appropriate source code from your program and a list of the flags you used.

3.2.1

Reporting Compiler Crashes: C and C++

Compile with the "-keep" option. This will cause the compiler to keep its intermediate files:

```
$ pathcc -O3 -c fred.c -keep
```

Send us the file "fred.i" - this is the result of preprocessing fred.c.

3.2.2

Reporting Compiler Crashes: Fortran

If your code is Fortran 77, you can send us the source file, the flags you used, and any include files. If your code is run through the C preprocessor, send us the file after preprocessing.

If your code is Fortran 95, also send us the appropriate .MOD files. We may also need to see the source code for the modules and the flags you used.

3.3

Compiler Error, but the Code is Correct

This category of bug happens when the PathScale compiler gives you an error, but you think the code might be correct. Determining what is a valid program is challenging, but we endeavor to do our best. Please follow these steps:

1. C/C++: Does the code work with gcc/g++?

The PathScale compilers use the same include files and system libraries provided by your Linux distribution. The source for most of the include files and libraries distributed with the PathScale C and C++ compilers are the ones from the appropriate version of GCC. The C and C++ parser used by our compiler is from the same GCC version. We have not tried to fix any of the language compliance or compatibility issues that may exist in those components.

Additionally, the acceptance of a source file by another compiler does not guarantee that the source complies with the language specification. There are very few compilers that strictly enforce the standard language definition.

2. Prepare an example that shows the bug, preferably as small as possible.
3. Consult your favorite language reference to double-check that the example really is standard-conforming.

For C/C++: Check to see how the code behaves with the same version of GCC from which the PathScale compilers are derived (for example, PathScale compiler version 3.1 is derived from GCC 4.2.0).

4. Send us the example.

3.4

Code Compiles and Links, But Gets the Wrong Answer

These bugs can be very difficult to fix, especially if they appear in large programs only when using IPA (inter-procedural analysis). Some of these bugs are not really compiler bugs at all; there are subtle programming mistakes, such as use of uninitialized variables, which only cause wrong answers at high levels of optimization.

We recommend these approaches to try to isolate the bug:

1. Try compiling the entire code with `-O0` (oh-zero), no optimization, to see if you get the expected answer. If not, you don't have much chance to find a successful work-around, and face extra work isolating the problem.
2. Turn down optimization on groups of source files, until you find a single file (or subset) that is being mis-compiled. For example, you can use binary-search: turn down optimization on the first half of the source files, then a fourth of the source files, and so on.
3. Add additional output statements to the program to find out where the code begins giving unexpected answers.
4. Use the information you've gathered so far to make a smaller example that shows the bug.

One challenge with bugs in this category is that they often disappear when anything changes; for example, adding a print statement will change how the variables printed are optimized.

3.5

Other Code Faster than PathScale Compiler's Code

If another 64-bit or 32-bit compiler produces an executable for your program that runs significantly faster than the code produced by the PathScale compiler, running on the same platform, then we consider that to be a bug. Small examples with profiling data (generated by the gnu tool `gprof`) are especially welcome.

3.6

Working Around Bugs

Reducing the optimization level can often make bugs disappear. Even if you do successfully work around a bug by reducing the optimization level, we do still

—

recommend that you file a bug report, so that the workaround will be unnecessary in future releases.

3.7

Release Dates and Bug Fixes

Once we've fixed a bug, our standard action is to incorporate the bug fix into the next release of the compiler, so that the bug fix can be fully tested against our entire regression suite. This may mean a delay before the bug fix reaches you. Please check our support web sites for the most up-to-date information on bugs and bugfixes. Support web pages are found on the PathScale web site.

<http://www.pathscale.com/support.html>

3.8

Proprietary or Confidential Source Code

Some of our customers have proprietary or confidential source code. We are willing to work with these customers to attempt to fix their bugs without source, or to keep their proprietary source confidential. For more details, please contact us at support@pathscale.com.

Index

B

Bugs
categories of 3-1

C

Compiler
executable runs slowly 3-5
gets a wrong answer 3-5
Compiler crash
C and C++ 3-4
example of 3-3
Fortran 3-4
problem report file 3-3
Compiler errors 3-4

G

gprof, using 3-5

P

pathbug 3-1
pathbug-helper 3-1, 3-2
pathhow-compiled 3-2
Proprietary source code 3-6

R

Releasing bug fixes 3-6

S

Support

email address 2-1
useful information to provide 3-2
web site urls 2-1

T

Trial license 2-1

