

TOTALVIEW NEW FEATURES GUIDE



VERSION 8.8



Copyright © 2007–2010 by TotalView Technologies. All rights reserved

Copyright © 1998–2007 by Etnus LLC. All rights reserved.

Copyright © 1996–1998 by Dolphin Interconnect Solutions, Inc.

Copyright © 1993–1996 by BBN Systems and Technologies, a division of BBN Corporation.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise without the prior written permission of TotalView Technologies.

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

TotalView Technologies has prepared this manual for the exclusive use of its customers, personnel, and licensees. The information in this manual is subject to change without notice, and should not be construed as a commitment by TotalView Technologies. TotalView Technologies assumes no responsibility for any errors that appear in this document.

TotalView and TotalView Technologies are registered trademarks of TotalView Technologies. TVD is a trademark of TotalView Technologies.

TotalView uses a modified version of the Microline widget library. Under the terms of its license, you are entitled to use these modifications. The source code is available at:

ftp://ftp.totalviewtech.com/support/toolworks/Microline_totalview.tar.Z.

All other brand names are the trademarks of their respective holders.

Contents



New Features: Versions 8.0–8.8

New Platforms and Compilers for Version 8.8	1
TotalView 8.8 Changes	1
Platform Changes in Previous Version 8 Releases	2
New and Changed Features	3
Version 8.7 Features	3
Version 8.6 Features	3
Version 8.5 Features	4
Version 8.4 Features	4
Version 8.3 Features	4
Version 8.2 Features	7
Versions 8.0 and 8.1 Features	7
Other Features	8

New Features:

Versions 8.0 to 8.8



This document contains information about changes made to TotalView for versions 8.0.0 through 8.8. While this information lets you know what changes have occurred, it doesn't completely describe these changes. More detailed descriptions can be found within the *TotalView Users Guide* and the *TotalView Reference Guide*.

New Platforms and Compilers for Version 8.8

TotalView supports new versions of platforms, compilers and environments. New platforms supported are Fedora 12 and Ubuntu 9.10. A new compiler is supported, PGI 10.1.

TotalView 8.8 Changes

Remote Display

A Remote Display client for the Mac and Windows 7 operating systems has been added.

Runtime Performance

Runtime performance at scale has been improved.

Platform Changes in Previous Version 8 Releases

- 8.7 Changes** TotalView supports the Fedora 9 and 10 platforms; compilers XLF 12.1, XLC 10.1, Intel 11.1, GCC 4.4, gfortran 4.4, and Berkeley UPC 2.8; and parallel environments MPICH 2.1 and OpenMP 1.3.
- 8.6 Changes** TotalView supports C/C++ compilers for the IBM Cell Broadband Engine, GNU Fortran from Red Hat, and the Sony BCU-100 Zego.
- 8.5 Changes** TotalView supports the IBM Cell Broadband Engine.
- 8.4.1 Changes** This release has updated the compilers TotalView supports. Consult the *TotalView Platforms and System Requirements Guide*. for more information.
- 8.4 Changes** TotalView supports Apple Mac OS X 10.5 (Leopard).
- 8.3 Changes** New operating system versions include:
- Apple OS X 10.4.5, 10.4.8, and 10.4.9
 - Fedora Core 7
 - Ubuntu 6.06
- As always, we have added support for new versions of existing compilers and parallel runtime environments. Consult the *TotalView Platforms and System Requirements Guide*. for more information.
- 8.2 Changes** TotalView 8.2 has added support for the following systems and compilers:
- SiCortex supercomputer
 - Cray XT4 support and APLs integration
 - Fedora Core 6
 - Expanded Mac support
 - Preliminary Mac OS X Leopard, 64-bit Mac-Intel, and Mac Universal Binaries
 - Ubuntu
 - Ubuntu is a community-developed Linux-based operating system for the desktop, laptop, thin client and server. TotalView will support applications developed on this platform.
- You'll find a complete list of supported platforms and compilers in the *TotalView Platforms and System Requirements Guide*.

New and Changed Features

Version 8.7 Features

This section lists the changes made for version 8.7 of TotalView.

- TotalView includes the MemoryScope GUI for memory debugging, with Red Zones on Linux platforms for detection and immediate notification of errors in heap arrays.
- Support for various heterogeneous debugging combinations and Power PC 32 cross debugging is introduced.
- Remote debugging is enhanced to support IP-only networks and nodes with multiple interfaces with different IP addresses.
- TotalView allows subset attach from the command line and via the CLI.
- New server launch CLI state variables and shared library search and mapping state variables are added, and the search path and mapping rules are changed.

Version 8.6 Features

This section lists the changes made for version 8.6 of TotalView.

- Remote Display: TotalView can open a window on your machine that displays TotalView executing on a remote system. We provide installers for Windows running XP or Vista, Linux x86 and Linux x86-64. While Remote Display can only run on these operating systems, the remote TotalView can execute on any platform that TotalView supports. Information on running Remote Display is at <http://www.totalviewtech.com/Documentation>
- TVScript / Batch debugging: TotalView can run unattended using the **tvscript** shell command. The commands that tvscript executes can be entered as command-line options or by creating a file for **tvscript** to execute. Chapter 4 of the *TotalView Reference Guide* explains how to use the **tvscript** command. This document is at <http://www.totalviewtech.com/Documentation>
- **Debug** is added to the TotalView menubar if you are running on Linux-x86 and Linux-x86-64. All memory commands are now within **Debug**.
- The current line is highlighted in yellow. If you have purchased a Replay license, an orange highlight line shows where you've gone back to. A separate marker shows the PC that existed when you entered replay mode.
- The **dhistory** command lets you invoke ReplayEngine from within the CLI. The **spurs** command displays information on spurs library use.
- Options supporting ReplayEngine are added to **dattach**, **dload**, and stepping commands such as **dstep**, **dnext**, **duntil**, etc.
- Options to the **dload** command let you start MPI programs. These options are **-mpi**, **-nodes**, **-starter_args**, **-np**, **-procs**, and **-tasks**.

- The **Process > Startup Parameters** dialog is rearranged and contains options for enabling memory debugging and ReplayEngine. This window comes up automatically when you start TotalView using a program's name as an argument.
- Other command-line options added for this release include `-local_interface` (most often used with Blue Gene), `-memory_debugging`, and `-replay`.
- New variables added at this release are `TV::ask_on_cell_spu_image_load`, `TV::cell_spu_image_ignore_regexp`, `TV::cell_spu_images_stop_regexp`, `TV::cell_spurs_jm_name`, `TV::cell_spurs_kernel_dll_regexp`, `TV::cell_spurs_ss_name`, `TV::cell_spurs_tm_name`, `TV::data_format_int128`, `TV::local_interface`, and `TV::GUI::heap_summary_refresh`.

Version 8.5 Features

No changes are listed for version 8.5 of TotalView.

Version 8.4 Features

This section lists the changes made for version 8.4 of TotalView.

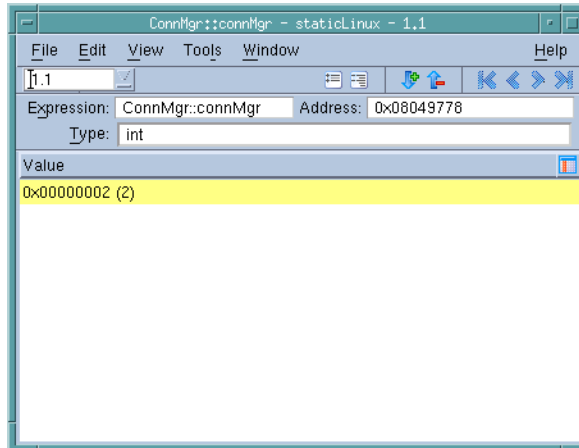
- If you have more than one TotalView license, you can control which kind of license TotalView uses by adding one of the following command-line options: `-team`, `-noteam`, `-teampplus`, `-noteampplus`, `-ent` or `-noent`. For more information, see Chapter 6 of the "*TotalView Reference Guide*."
- The TotalView Memory Debugger can now write light-weight memory debug files when an event occurs. These files are similar to the memory debugging files (`.mdbg`) files that you can write using the **File > Export** command. They differ in that they are designed to be written when the event occurs and in such a way that the program's behavior is minimally disturbed. These files are described in the Chapter 3 of the "*Debugging Memory Problems with TotalView*" Guide.
- Improved support for C++ templates.
- Improved support for Fortran modules on Apple Mac OS X.

Version 8.3 Features

This section lists the changes made for version 8.3 of TotalView.

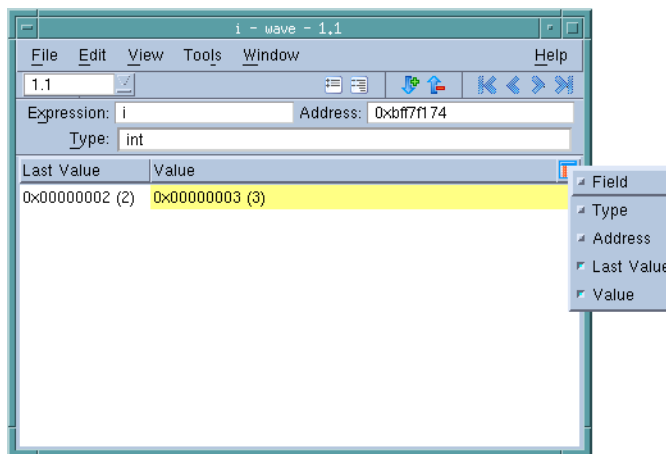
- Improvements to the way TotalView launches MPI programs let you use TotalView with virtually every MPI library, even with those that were not configured for debugging.
- TotalView now highlights changes to values displayed in the **Variable** and **Expression List** windows with a colored background. (See Figure 1 on page 5,) While this figure shows a simple variable, TotalView also highlights changed elements within compound variables such as structures and arrays.
- Values in the **Variable** and **Expression List** windows have a **Previous value** hidden column that you can display. Use the control on the right side of

Figure 1: Highlighted Change in the Variable Window



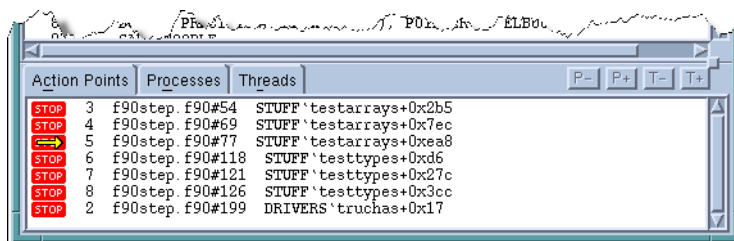
the column headings to display a list of columns that you can display or hide. (See Figure 2 on page 5,)

Figure 2: Last Value Column



- When a process hits a breakpoint, TotalView highlights the breakpoint by placing an arrow over the breakpoint ID in the Action Points pane. (See Figure 3 on page 5,)

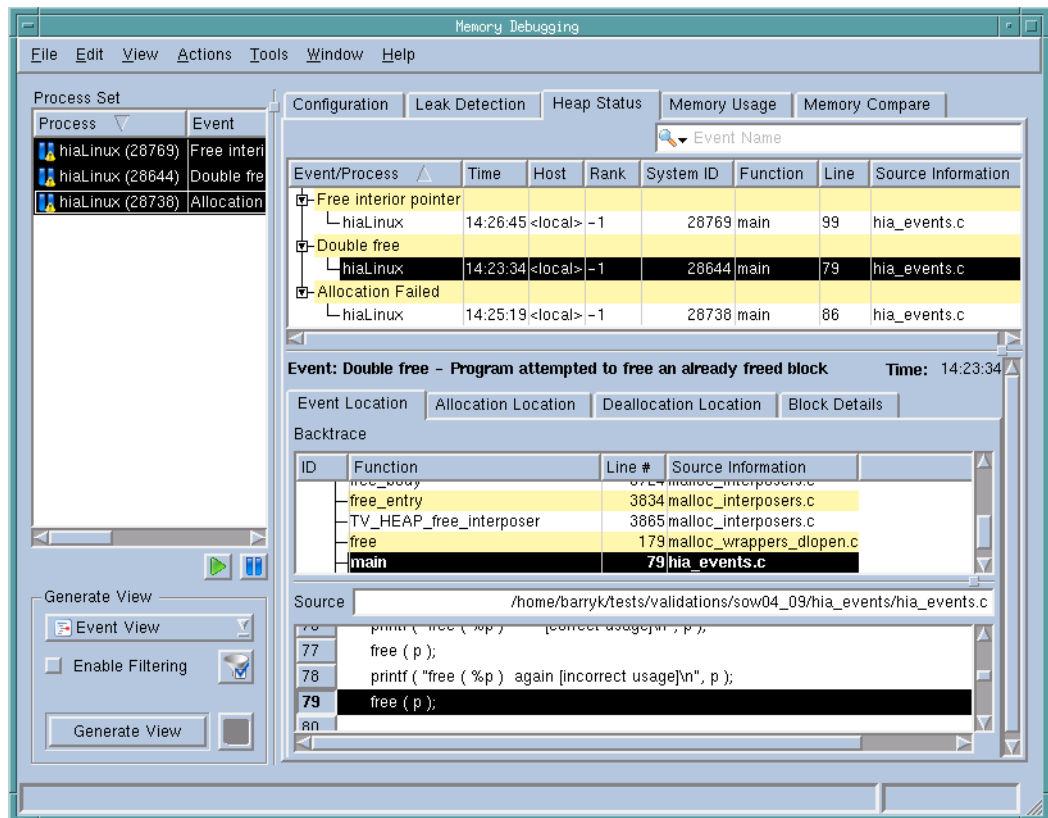
Figure 3: Highlighted Action Point ID



- View > Show Across replaces View > Laminate in the Variable window's menus. This means the commands you will now use are View > Show Across > Process and View > Show Across > Thread.

- You can now tell TotalView to show a variable across processes or threads by right-clicking on it in the Source Pane, then selecting either **Across Processes** or **Across Threads** from the context menu.
- The **Create Watchpoint** command was added to the Action Points menu. As always, you can create a watchpoint from within the Variable window by selecting **Tools > Create Watchpoint**.
- You can now set a watchpoint upon a variable's memory address by right-clicking on the variable in the Source pane and then selecting **Create Watchpoint** from the context menu.
- You can completely expand or collapse information in the Variable window by selecting an icon in the toolbar. The accelerators for these commands are **Ctrl++** (that's the control key and the + symbol) and **Ctrl+-** (which is the control key and the - symbol).
- TotalView no longer stops by default when your program loads a library.
- You can specify more than one core file on the command line and you can use wildcards in core file names.
- There is a new Events View report within the Memory Debugger Heap Status tab.

Figure 4: Event View



- Within the Memory Debugger's **File > Import Data** dialog box, you can select multiple memory debugging (.mdbg) files.

Version 8.2 Features

This section looks at changes that have occurred within TotalView.

■ Early-Access GUI Installer

You can now install TotalView from tar files as you've always done or install it using our new graphical installer. We are calling this an early-access release in that we want you to tell us what you think of it and how we can improve it.

■ Fortran Parameter Display

TotalView now displays the value of Fortran parameters. Parameters can be used like variables in expressions but could not previously be examined within the debugger.

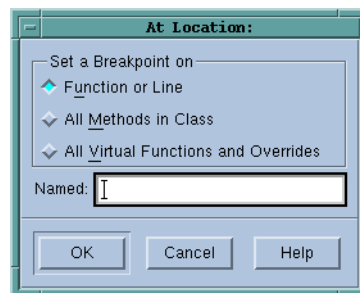
Versions 8.0 and 8.1 Features

Breakpoint Changes

Action points are considerably more powerful. Here is a summary:

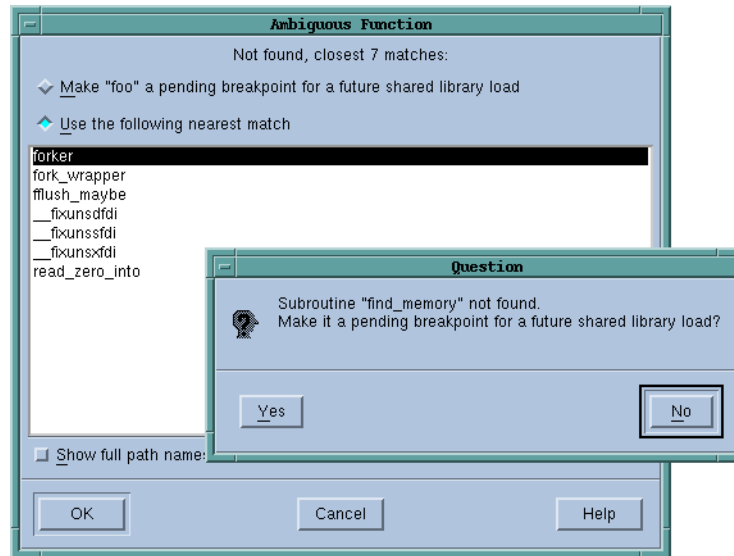
- The **Action Point > At Location** command now has three choices:
 - **Function or Line:** lets you set a line number or a function name. This choice is what occurred in previous TotalView releases.
 - **All Methods in Class:** lets you set a breakpoint on all methods in a class. This can set more than one breakpoint.
 - **All Virtual Functions and Overrides:** lets you set breakpoints on virtual functions and their overrides. This too can set more than one breakpoint.

Figure 5: Breakpoint > At Location Dialog Box



- You can now tell TotalView that a breakpoint will occur in a library that will be loaded later and that TotalView should retain knowledge of this breakpoint. This allows it to be set when the library is read. Previously, TotalView had to create and set breakpoints at the same time. This meant that when you enter a name into the **At Location** dialog box and the name is not yet known, TotalView, displays either an **Ambiguous Function** or a **Question** dialog box. At this time, you can set the breakpoint's status to *pending*. (See Figure 6 on page 8.)
- When you create a barrier breakpoint or change a breakpoint to a barrier point, the same new features are available.
- The CLI **dbreak**, **dbarrier**, and **dlist** commands have been extended to use these features. The argument to these commands can now be a break-

Figure 6: Setting Pending Breakpoints



point expression. Understanding this concept reveals some of the subtleties involved using these new features. These concepts are explained with the **dbreak** pages of the *TotalView Reference Guide*.

Other Features

This section describes improvements and changes made in many different places in TotalView.

- The way in which you set search paths has changed. (See Figure 7 on page 9.)
You have told us that most of the time, you usually do not need to set search paths. If you do, you just need to enter a couple of paths. This is what the old dialog box was designed for. (You can still do this by typing paths directly into the **EXECUTABLE_PATH** tab.) However, when your programs make the transition from modules being developed on your workstation to an place where the work of development teams is brought together, setting search paths was tedious and difficult to get right. This new dialog box, extensively documented in the help, lets you solve this problem.
- The **File > New Program** dialog box has changed. Much of what you will see makes the dialog box more usable. Most notable is the way you attach to processes. The dialog box now lets you select more than one process at a time. The one new feature is that you can now enable memory debugging from this dialog box.
- The **View > Freeze** command is added to the Variable window. This command tells TotalView that it should freeze the contents of a Variable window. That is, as your program executes and as data values change, the contents of this window does not change. In most cases, you will also create a second Variable Window so that you can see old and new values at the same time.

Figure 7: File > Search Path Dialog Box

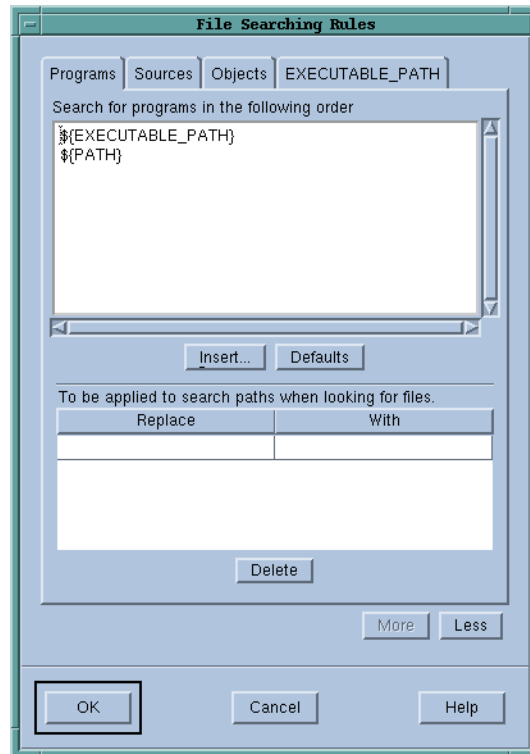
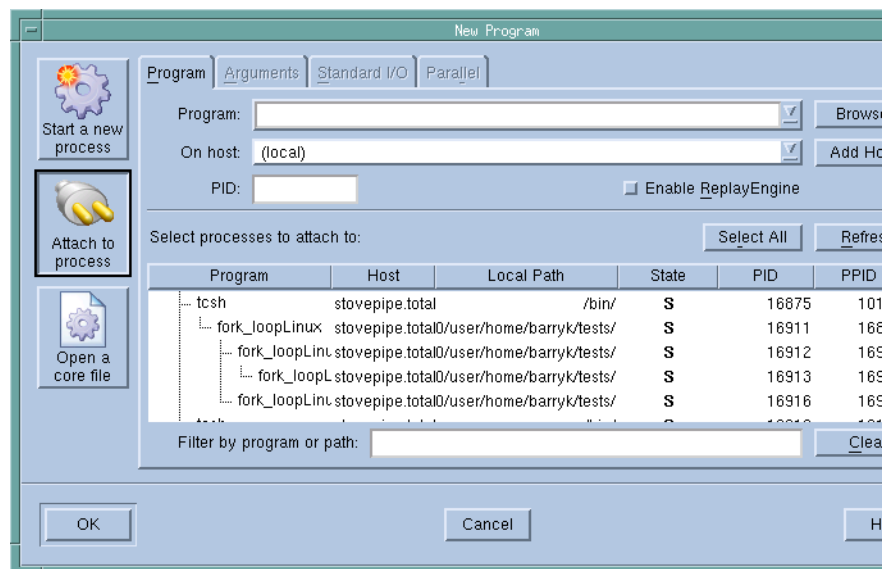


Figure 8: File > New Dialog Box



- The **View > Lock** command is added to the Variable Window. This command tells TotalView that it should not change the address from which the Variable Window is obtaining information.
- New **dheap -compare** CLI command options. This options lets you compare the result of two different memory states.

- New **dkill --remove** CLI command option. Using this option to tell TotalView that, in addition killing the process, it should remove knowledge of the process. This is seldom necessary. However, if you are using TotalView Team, using this option makes the token used by a process available to another process in your program.
- The following CLI variables were added for this release
 - **TV::env**: sets an environment variable.
 - **TV::bluegene_server_launch_string**: sets the Blue Gene server launch string.
 - **TV::default_sterr_append**: tells TotalView to append **stderr** information to **stdout**.
 - **TV::default_stderr_filename**: tells TotalView to write **stderr** information to a file.
 - **TV::default_stderr_is_stdout**: tells TotalView to write **stderr** information to **stdout**.
 - **TV::default_stdin_filename**: Tells TotalView to read **stdin** information from a file
 - **TV::default_stdout_append**: Tells TotalView to append **stdout** information to a file
 - **TV::default_stdout_filename**: Tells TotalView to write **stdout** information to a file.