TotalView 2016.07 Release Notes

Updated: November 2016

These release notes contain a summary of new features and enhancements, late-breaking product issues, migration from earlier releases, and bug fixes.

PLEASE NOTE: The version of this document in the product distribution is a snapshot at the time the product distribution was created. Additional information may be added after that time because of issues found during distribution testing or after the product is released. To be sure you have the most up-to-date information, see the version of this document on the Rogue Wave web site:

http://www.roguewave.com/support/product-documentation/totalview.aspx

Additions and Updates

Early Access to the Next Gen TotalView User Interface

The TotalView team continues to add more features to the next generation TotalView user interface. The interface is in its early feature stages but we are excited to provide an early version so that we can gather feedback from TotalView users on its initial capabilities. To try out the new user interface start TotalView with the -newUI switch:

```
totalview -newUI
```

Check out the in product help through the Help | Contents menu item for more details about the new user interface. New features added to the NextGen user interface in this release include:

- Fortran debugging support
- New platform support for Linux ARM64 and Apple's macOS/Mac OS X

Currently, the next generation UI is only supported on Linux x86 64-bit, Linux PowerLE, Linux ARM64 and Apple's macOS/Mac OS X platforms. It supports multi-process and multi-threaded debugging and also supports a level of parallel, MPI and CUDA debugging. Creating evaluation, barrier and watch points through the new UI is not supported yet but evaluation and barrier points will show up in the UI properly if created through the CLI. Release over release we will be adding new functionality based on a priority list that you can help influence. Please send email to tv-

beta@roguewave.com with your feedback and feature priorities! We welcome all feedback and feature requests for the new user interface!

Early Access Support for Linux ARM64

Early Access support for Linux ARM64 has been added with the TotalView 2016.07 release! Most functionality of the debugger is supported including memory debugging with MemoryScape and the new TotalView user interface. We have tested on Cavium's ARM64 processor running Ubuntu 14.04 and NVIDIA's Jetson TX1 running Ubuntu 16.04. We encourage you to give TotalView a try on your Linux ARM64 system and report any issues (and successes!) to tv-beta@roguewave.com. The following are the known limitations and issues:

- Hardware data watchpoints are not supported on NVIDIA Jetson TX1. Setting a watchpoint in the debugger will crash the Linux kernel.
- TotalView cannot unwind stack frames that have no debug information
- TotalView has limitations debugging user level threads and does not support asynchronous thread control
- Reverse debugging with ReplayEngine is not supported yet

CUDA 8 Support

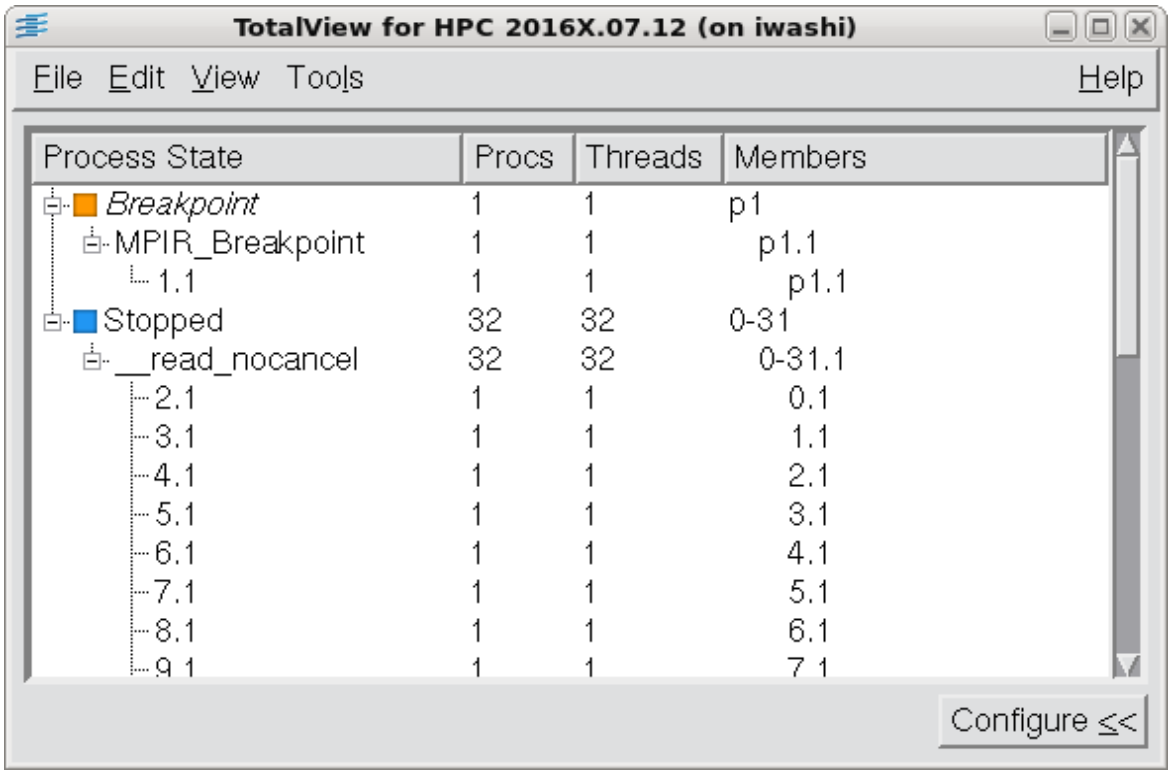
TotalView 2016.07 has been validated against the official CUDA 8 release.

macOS Sierra Support

The TotalView 2016.07 release adds support for Apple's macOS Sierra. It has also been tested against the latest macOS Sierra update, 10.12.1, and XQuartz 2.7.11. We have found that with versions 2.7.9 and newer of XQuartz, the TotalView Visualizer is not working. We'll be analyzing what changed starting in version 2.7.9 and plan to implement a fix for the next TotalView release.

Root Window Enhancements

TotalView's Root Window displays grouped threads and processes using common properties and allows you to quickly see the state of all the processes and threads in your debugging session. For this release, color coded icons have been added to the Root Window to help you quickly distinguish between the different process and thread states.



Bug fixes and improvements

There have been a significant number of bug fixes and improvements added to the 2016.07 release including:

- Improved stepping and debugging through inlined functions
- Significant startup and stability improvements when debugging parallel CUDA jobs
- Performance and stability enhancements for TotalView's reverse debugging feature, ReplayEngine.

Platform Updates

TotalView 2016.07 introduces support for the following platforms

- **Platforms**:

  Fedora 24

  Ubuntu 16.04

  macOS Sierra

- **Compilers**:

  Intel Composer XE 2017 (17.0)

  GCC 6.2

- **Parallel:**

  OpenMPI 2.0.0

  Intel MPI 2017

Bug Fixes for 2016.07

TVT-19959 Group Go while debugging a parallel program in record mode fails to advance off a breakpoint

TVT-21165 TotalView steps into inlined code rather than 'Next'ing over it

TVT-21316 TotalView exhibits crashes and severe performance issues with Replay when stressing memory overhead.

TVT-21343 CPU usage by TotalView makes debugging increasingly unusable as the debugging session progresses.

TVT-21556 Can't Dive into "this" for some objects compiled by gcc on AIX.

TVT-21576 Fatal Error when connecting to manually launched debug server.

TVT-21577 TotalView crashes when a window is closed using the red button on Mac OS X El Capitan.

TVT-21583 TotalView session disappears when using Kill button with an MPI program launched from the GUI.

TVT-21640 Fix C++11 demangling to properly handle names assigned to lambda function bodies and other complex template objects.

TVT-21682 TotalView does not indicate when the -newUI flag is unsupported for a platform.

TVT-21734 Clarify 'Create a flexlm user' instructions in Configure_License.

TVT-21785 Target program doesn't load from command line when TVNEWUI environment variable is set.

TVT-21862 CUDA qualifier base name is lost when converting "char" to "$string" type.

Deprecation Notices

CLI *replay* command's –get_time and –go_time deprecated

The following two CLI arguments for the *replay* (which is an alias for *dhistory*) command have been deprecated

-get_time - display the current time

-go_time - put the process back to the specified virtual time

The arguments will still be available in this release but will be removed in a subsequent release. A warning will be displayed when the arguments are used. In place of these arguments, we recommend using the new CLI Replay bookmark facility instead, which provides much better accuracy in returning you to the exact point in Replay history.

Through the Replay bookmark facility, it is now possible to create a bookmark at any point in recorded history and then go back to that exact point at any time. The Replay bookmarking facility is accessed through the CLI's *replay* command through the following new arguments:

- -create_bookmark *[comment]*
  Create a replay bookmark at the current location. You can specify an optional 'comment' to this command and it will be stored with the bookmark for display purposes. A bookmark will be created with a unique numeric 'ID'.
- -delete_bookmark *ID*
  Delete the bookmark with the specified ID.
- -clear_bookmarks
  Delete all Replay bookmarks.
- -goto_bookmark *ID*
  Go to the bookmark with the specified ID. This will bring the focus process back in time to the place where the bookmark was first created.
- -show_bookmarks
  Display all Replay bookmarks. This command shows the bookmark ID along with information about what line number, pc and function the bookmark is on.

  Example:

  ```
  dhistory -show_bookmarks
  ```

```
bookmark: 1:  pc: 0x<Address>, function: main, line: 59, comment:
Starting bookmark
bookmark: 2:  pc: 0x<Address>, function: main, line: 60, comment:
bookmark: 3:  pc: 0x<Address>, function: main, line: 69, comment:
```

Known Issues

Licensing

TotalView sometimes cannot acquire license due to FlexNet bug

If you are using Linux Power, Linux Itanium or AIX then you are still affected by the bug in the FlexNet Publisher software that results in TotalView's inability to acquire a license when your license file contains multiple licenses with different maintenance expiration dates (i.e. the 4th field on the INCREMENT line). The licensing software skips some of the licenses in this case. If you know that your license file is being read and is correct, and you think you might be running into this bug, we recommend that you add the "sort" keyword and value (such as sort=1, sort=2, sort=3) to each INCREMENT line in the license file in any order. This bug has been reported to Flexera and is identified as SIOC-000145042.

Here is an example of adding the "sort" keyword:

```
SERVER linux-power 0050569b402c
VENDOR toolworks
INCREMENT TotalView_Enterprise toolworks 2014.1231 permanent 1 \
        sort=1 VENDOR_STRING="processors=16 platform=linux-power" \
        SIGN=3350A26C395A
INCREMENT TotalView_Enterprise toolworks 2015.1231 permanent 1 \
        sort=2 VENDOR_STRING="processors=16 platform=linux-ia64" \
        SIGN=C7D11FB667C8
INCREMENT TotalView_Enterprise toolworks 2016.1231 permanent 1 \
        sort=3 VENDOR_STRING="processors=16 platform=linux-x86_64" \
        SIGN=BEC7534A248A
```

Linux ARM64 and Linux PowerLE use an Expiring Licensing Model

The Linux ARM64 and Linux PowerLE platforms use an expiring license model since neither are supported by the FlexLM license manager. TotalView will cease to run after six months of the 2016.07 release date. Subsequent releases will extend the expiration date or provide a FlexLM license managed solution.

macOS

With multiple displays attached to a macOS machine, some TotalView windows may not be noticeable

When a user attaches an external monitor to a running Macbook Pro, or adds multiple displays to a Mac, window rearrangement may move some windows off-screen. This may result in a TotalView modal window not being found until you use the Mac command to display all the windows (Mission Control). This appears to be an interaction between XQuartz and Darwin. It has been seen in Mavericks, but it's possible it will show up in other releases. There may be a workaround in System Preferences->Mission Control by disabling "Displays have separate Spaces."

Physical console access needed when starting TotalView

Starting in Mountain Lion, OS X security policies require that users meet a password challenge in order to use TotalView, and the challenge can be issued only to the console (the OS X desktop). After the password challenge is met once, you can run TotalView repeatedly from the same login session without further challenges.

It is possible to work around this need for physical access with the following steps. The first few of these are likely already set in order to allow TotalView to run.

- Install XQuartz and TotalView
- Ensure every user needing debugging is in the _developer group
- Allow X11 forwarding in the sshd_config file (disabled by default)
- In a terminal window enter the following two commands:
  - DevToolsSecurity -enable (this step is optional if this was already enabled)
  - sudo security authorizationdb write system.privilege.taskport allow

TotalView Remote Display Client hangs

If you are using the Remote Display Client on Mac OS X Mountain Lion or Mavericks, the application may freeze if more than one Terminal window is open. To get around this issue, set the display number in the Terminal's Advanced Options dialog to a value that does not conflict with another user.

Visualizer fails under macOS Sierra and Xquartz

Attempts to use the visualizer tool fails with a message "Error: attempt to add non-widget child "dsm" to parent "vismain" which supports only widgets.

Linux

OpenMPI 1.8.4 with ReplayEngine enabled on older Linux releases.

We have observed a problem with the combination of Replay Engine, Open MPI 1.8.4, and older Linux releases such as RHEL5 (Red Hat Enterprise Linux). When the MPI runtime system closes shared libraries during its startup, a munmap(2) system call may attempt to unmap memory which is in use by Replay Engine. The error message "Unsupported memory access with syscall (11). Conflict with replay private internal memory." is displayed. This error is unrecoverable, but it may be possible to use Replay Engine on the same application by enabling it after the application has completed its MPI_Init call. We have not seen this problem with newer Linux releases such as RHEL6.

TotalView Message Queue and Intel MPI 5.0

By default, the TotalView Message Queue will not work with Intel MPI 5.0 without setting correctly LD_LIBRARY_PATH to the Intel MPI debug libraries. This can be done by sourcing one of the "mpivars.sh/csh" scripts provided by Intel with an added "debug" argument. For example, issue the command "source PATH/impi/5.0.3.048/bin64/mpivars.sh debug", making sure to replace PATH with the path to your Intel MPI compiler installation. TotalView will then properly pick up the MPI message queue information and display it in its Message Queue window.

Memory Debugging and Intel MPI 5.0+

If a user wants to do memory debugging and they statically link their MPI program with the Intel MPI 5.0+ libraries, MemoryScape will detect a Double Allocation error. This is because, starting with Intel MPI 5.0, the MPI libraries redefine free() and MemoryScape depends on the system free() to see the deallocations. To work around this problem, the user will need to link dynamically or fall back to the Intel MPI 4.0+ libraries.

Debugging IBM Platform MPI Jobs in TotalView

Users have seen some issues when trying to use TotalView on an IBM Platform MPI job. If you try to launch the job from the Session Manager, or the Parallel Tab of the Startup Parameters window, TotalView may show an error that the target program has crashed while trying to load shared libraries. When run under mpirun, this error does not show since mpirun sets up the environment correctly. One can avoid the problem by setting the environment variable LD_LIBRARY_PATH to add the path to the library directory containing the missing libraries. The libraries should be in the 'lib' directory that is the same level as the 'bin' directory containing mpirun.

While testing the above issue it was noted that the classic launch method of

```
totalview  mpirun –a –np 4 ./foo
```

did not appear to work correctly. TotalView would attach to all the processes, but only rank 0 was held at the point where the job went parallel. The other processes would run to a point where they were waiting on rank 0. To work around this, launch through the GUI as described above, or use the –tv switch for mpirun

```
mpirun –tv –np 4 ./foo
```

Newer Linux kernels that prohibit non-root access to /proc/self/pagemap and ReplayEngine

If non-root access to /proc/self/pagemap is prohibited, the ReplayEngine will emit an ignored assertion warning when the program being debugged enters record mode for the first time. Furthermore, any unknown syscalls will subsequently be handled a little more slowly.

The change affects Ubuntu 15.04 and likely other new distribution releases.

While using ReplayEngine, attaching to 32-bit application from 64-bit hosts sometimes fails

On some 64-bit hosts, attaching to a 32-bit target fails and results in a crash. The underlying technology behind ReplayEngine assumes that in a 64-bit environment, the target is also a 64-bit application and was not explicitly designed to support a mixed environment.

Benign Warning Messages Displayed when ReplayEngine is run on SuSE Linux Enterprise Server 11 with Service Pack 1. (SLES 11 SP1)

As of the 2016.06 release, ReplayEngine no longer fails when attempting to do replay mode operations (move the target backward into history) on Linux x86-64 platforms running SuSE Linux Enterprise Server 11 with Service Pack 1 but some warning messages such as the following are displayed:

```
127083  client/set_current_child.c:456:set_current_child_fn
[78347:78347]: Failed to restore process name for pid 78357: -5

127084  client/set_current_child.c:179:set_current_child_fn
[78347:78347]: Failed to set process name for pid 78357: -5
```

These messages are benign and your reverse debugging session will work normally.

We have only observed this problem on SLES 11 SP1. It is possible however, that other platforms running the same Linux kernel version (2.6.32.12-0.7) will also run into this problem. The only available workarounds are to update the OS to a more recent release (for example, installing Service Pack 2).

Linux - Ubuntu

Memory debugging by linking against the TotalView libraries may not work

There are a number of cases in which it is recommended to link the Heap Interposition Agent (HIA) into the target program to allow memory debugging without having to enable it in the GUI each time. Starting in Ubuntu 12, the linker does not link in libraries that are not directly used by the program. This means that the link line for the agent, with TVLIB pointing to the TotalView library,

gcc –g –o memprog memprog.c –L$TVLIB –ltvheap –Wl,-rpath,$TVLIB

may not pick up the HIA. Instead, add the option "-Wl,--no-as-needed" before the inclusion of the tvheap library. The new compile/link line will look like

```
gcc -g -o memprog memprog.c -Wl,--no-as-needed -L$TVLIB -ltvheap -Wl,-rpath,$TVLIB
```

std::string shows as opaque value compiled with Clang 3.5

When trying to debug a program compiled with Clang 3.5 that uses a std::string variable, the variable is listed as type std::string:64 with a value of

Opaque std::basic_string<char,std::char_traits<char>,std::allocator<char>>

When the same program is compiled with the Clang 3.3 compiler, the std string is seen as a simple STL container, and the string value is seen as 'abcd'.

Clang supports a number of optimizations to reduce the size of debug information in the binary. These optimizations work based on the assumption that the debug type information can be spread over multiple compilation units. For instance, Clang does not emit type definitions for types that are not needed by a module and could be replaced with a forward declaration. Further, Clang only emits type info for a dynamic C++ class in the module that contains the vtable for the class.

It has been found that using the -fstandalone-debug option which turns off these optimizations works around the problem with the opaque value above.

The -fstandalone-debug option is useful when working with 3rd-party libraries that don't come with debug information.

Note that Clang never emits type information for types that are not referenced at all by the program.

-fstandalone-debug is the default on macOS.

-fno-standalone-debug is the default on Linux-x86-64. To work around the opaque value problem above, use the -fstandalone-debug option.

CUDA

Dynamic parallelism not fully supported

With CUDA 5.5-7.0, we have limited support for dynamic parallelism. You should be able to use TotalView in the CUDA 5.5/CUDA 6.5 runtime with applications that display dynamic parallelism; however, we plan improvements to our functionality for displaying the relationships between dynamically launched kernels and navigating the various running kernels.

Layered textures not supported

TotalView does not yet support CUDA layered textures. If you try to examine a layered texture in the TotalView Data Pane, a "Bad address" message will be displayed and you will see "ERROR: Reading Texture memory not currently supported" displayed on the console. If you require layered textures support, please contact TotalView support at support@roguewave.com and let us know how you are using textures so we can develop the best solution to support you.

SGI

Memory debugging MPI programs on SGI systems needs special linking

The TotalView and MemoryScape memory debugging Heap Interposition Agent (HIA) technology conflicts with the SGI memory manager when used in MPI programs. The easiest way to get around this problem is to disable the SGI memory manager by unsetting the MPI_MEM_ALIGN environment variable. Without this variable set, the SGI memory manager will not be loaded and the HIA will work correctly, enabling memory debugging to take place.

Cray

Debugging your program within supercomputer environments can often be challenging. Reference the sections below to learn pointers on how to successfully enable memory debugging, and perform reverse debugging on your program within a Cray environment.

Memory debugging on Cray systems

Use the pointers below to help achieve a successful memory debugging session within the Cray environment:

- Install TotalView on a shared file system visible to the Cray compute nodes.
  In order for the required memory debugging shared libraries to be located when your program is running on a compute node it is best to install TotalView on a shared file system.
- Cray TotalView Support Module is not required for TotalView 8.15.0.
  As of TotalView 8.15.0 and its use of the MRNet tree framework TotalView no longer requires the Cray TotalView Support Library to be installed in order to run. If the MRNet tree framework is turned off then the Cray TotalView Support Module will be required.

- Statically linking your program against the tvheap_cnl library.
  One of the most foolproof ways of enabling memory debugging is to statically link against the tvheap_cnl library that is shipped with TotalView. The static library fully supports multithreaded applications. Statically linking your application with the tvheap_cnl library will automatically enable memory debugging in your program when debugged under TotalView and MemoryScape. See the "Linking Your Application with the Agent" discussion in the User Guide for more information on how to statically link your applications with the library.

- Dynamically linking your program against the tvheap_cnl library.
  It is possible to dynamically link your application against the dynamic version of the tvheap_cnl library. In this scenario the tvheap_cnl library must be visible to the Cray Compute Node systems, either through a shared file system or by the Cray environment automatically staging the applications shared library dependencies on the compute node.

- Do not enable memory debugging on the aprun starter process.
  Turning on memory debugging for the aprun starter process will cause it to fail and prevent the job from starting. The proper way to enable memory debugging is to use the static or dynamic linking options described above.

Reverse debugging on Cray systems

Use the pointers below to help achieve a successful reverse debugging session within the Cray environment:

- Do not enable reverse debugging on the aprun starter process.
  Turning on reverse debugging for the aprun starter process will cause it to fail and prevent the job from starting. The proper way to turn on reverse debugging is to launch your parallel job and reach a breakpoint in the code and then dynamically turn on the Replay Engine reverse debugging option. It will begin recording the execution of the program from that point forward.